© Copyright 2005 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG.
The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C* are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

Disclaimer
Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressively prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP’s Support Services and may not be modified or altered in any way.

Documentation in the SAP Service Marketplace
You can find this documentation at the following Internet address:
service.sap.com/securityguide
### Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Text</td>
<td>Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Cross-references to other documentation.</td>
</tr>
<tr>
<td>Example text</td>
<td>Emphasized words or phrases in body text, graphic titles, and table titles.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.</td>
</tr>
<tr>
<td>Example text</td>
<td>Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td>Example text</td>
<td>Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td>&lt;Example text&gt;</td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td>EXAMPLE TEXT</td>
<td>Keys on the keyboard, for example, F2 or ENTER.</td>
</tr>
</tbody>
</table>

### Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Caution</td>
</tr>
<tr>
<td>📖</td>
<td>Example</td>
</tr>
<tr>
<td>📝</td>
<td>Note</td>
</tr>
<tr>
<td>🟢</td>
<td>Recommendation</td>
</tr>
<tr>
<td>🔄</td>
<td>Syntax</td>
</tr>
</tbody>
</table>

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see Help on Help → General Information Classes and Information Classes for Business Information Warehouse on the first page of any version of SAP Library.
Contents

SAP NetWeaver Application Server Java Security Guide ................. 3

1 Before You Start ................................................................. 3

2 Technical System Landscape .................................................. 3

3 User Administration and Authentication ..................................... 3

3.1 User Administration and Standard Users .................................. 3

3.1.1 User Administration Tools ............................................... 3

3.1.2 User Types ................................................................. 3

3.1.3 Standard Users .......................................................... 3

3.1.4 Standard User Groups .................................................. 3

3.2 User Data Synchronization .................................................... 3

3.3 Authentication Mechanisms and Single Sign-On Integration .......... 3

3.3.1 Declarative and Programmatic Authentication .......................... 3

3.3.2 Login Modules and Login Module Stacks ................................ 3

3.3.3 Authentication Schemes .................................................. 3

3.3.4 Integration in Single Sign-On Environments ............................ 3

4 Authorizations ....................................................................... 3

4.1 Standard Roles ............................................................... 3

4.2 Permissions, Actions, and UME Roles ...................................... 3

4.3 Standard UME Actions ....................................................... 3

5 Network Security .................................................................... 3

5.1 Transport Layer Security ..................................................... 3

5.2 Communication Channel Security ......................................... 3

5.2.1 Using an Intermediary Server to Connect to the J2EE Engine ......... 3

5.2.2 Communication Security for the Web Container ....................... 3

5.2.3 Communication Security for the EJB Container ..................... 3

5.2.4 Communication Security for Web Services ............................ 3

5.2.5 Communication Security for Persistency Stores ..................... 3

5.2.6 Communication Security for Software Deployment .................. 3

5.3 J2EE Engine Ports ................................................................ 3

6 Data Storage Security ........................................................... 3

7 Dispensable Functions with Impacts on Security .......................... 3

8 Other Security Relevant Information ........................................ 3

8.1 Security on JMS Service ..................................................... 3

8.2 Java Virtual Machine Security ............................................. 3

8.3 Security Aspects for the Database Connection ........................... 3

8.4 Working with the SDM ....................................................... 3

8.5 Destination Service .......................................................... 3
9 Tracing and Logging..................................................................................3
9.1 UME Logging and Monitoring .................................................................3
9.2 Security Aspects When Using HTTP and Web Container Tracing.....3
9.3 Masking Security-sensitive Data in the HTTP Access Log...........3
10 Virus Protection and SAP GUI Integrity Checks .........................3
SAP NetWeaver Application Server Java Security Guide

Purpose
This guide is intended to provide you with an overview of the security aspects and recommendations that apply for the SAP NetWeaver Application Server (SAP NetWeaver AS) for Java Server technology.

The J2EE Engine is the primary engine for the usage type Application Server Java (AS-Java) of the SAP NetWeaver. Therefore, the security aspects and recommendations for the AS-Java are equally relevant to securing the J2EE Engine.

Target Audience
- Technology consultants
- System administrators

This guide is not included as part of the Installation Guides, Configuration Guides, Technical Operation Manuals, or Upgrade Guides. Such guides are only relevant for a certain phase of the software life cycle, whereby the Security Guides provide information that is relevant for all life cycle phases.

Why Is Security Necessary?
With the increasing use of distributed systems and the Internet for business transactions and business data management, the demands on security are also on the rise. When using a distributed system, you need to be sure that your data and processes support your business needs without allowing unauthorized access to critical information. User errors, negligence, or attempted manipulation on your system should not result in loss of information or processing time. These demands on security apply likewise to the usage type AS-Java of the SAP NetWeaver platform. To assist you in securing the AS-Java, we provide this Security Guide.

Integration
There is also an SAP NetWeaver Application Server ABAP Security Guide [SAP Library].
About this Document

This security guide provides an overview of the security-relevant information that applies to the AS-Java. It contains an overview of the security considerations for the AS-Java and links to the security administration or development functions in the J2EE Engine Administration and Development Manuals, respectively.

The Security Guide contains the following sections:

- **Before You Start [Page 3]**
  Provides links to additional information, a list of important SAP Notes and other security guides that apply to securing the J2EE Engine.

- **Technical System Landscape [Page 3]**
  Provides a brief overview of the technical system landscape of the Java systems.

- **User Administration and Authentication [Page 3]**
  Describes user management, standard user types and synchronization of user data, as well as, AS-Java authentication mechanisms and Single Sign-On integration.

- **Authorizations [Page 3]**
  Provides an overview of the authorization concepts on the J2EE Engine. The topics discussed include authorization checking on the J2EE Engine, standard User Management Engine (UME) actions and security roles.

- **Network Security [Page 3]**
  Provides an overview of the communication channels used by the J2EE Engine and the corresponding transport layer security mechanisms. We also provide an example of a secure network infrastructure using network zones and information on the standard communication ports used by the J2EE Engine.

- **Data Storage Security [Page 3]**
  Describes the aspects in maintaining the availability, confidentiality and integrity of security sensitive data stored and used by the J2EE Engine.

- **Dispensable Functions with Impacts on Security [Page 3]**
  Provides information about deactivating optional J2EE Engine services that you may not need in productive operations.

- **Other Security Relevant Information [Page 3]**
  Presents an overview of additional topics relevant to securing the J2EE Engine, such as Java Virtual Machine (JVM) security, security of the JMS service, Database connection security and security for the Software Deployment Manager (SDM).

- **Tracing and Logging [Page 3]**
  Provides a discussion of the security aspects in using the logging and tracing functions available on the J2EE Engine.

- **Virus Protection and SAP GUI Integrity Checks [Page 74]**
  This section provides information on virus protection using the virus scan interface as well as information about how SAP GUI ensures its own integrity using checks.
# 1 Before You Start

## Fundamental Security Guides

The open architecture of the J2EE Engine gives you the flexibility to use the AS-Java for a number of functions in your system landscape. To perform the functions that you designate it, the AS-Java can communicate with a number of partners in your system landscape. Therefore, the corresponding Security Guides for SAP NetWeaver also have implications for securing the AS-Java.

Pay particular attention to the most relevant sections as indicated in the table below.

### Fundamental Security Guides

<table>
<thead>
<tr>
<th>Security Guides</th>
<th>Most-Relevant Sections</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP NetWeaver</td>
<td>Technical System Landscape [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>User Administration and Authentication [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Network and Communication Security [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guides for SAP NetWeaver According to Usage Types [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guides for Standalone Engines and Tools [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guides for Connectivity and Interoperability [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guides for the Operating System and Database Platforms [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Aspects for System Management [SAP Library]</td>
</tr>
<tr>
<td>SAP NetWeaver Usage Types</td>
<td>Security Guide for Usage Type AS [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guides for Usage Type EP [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guide for Usage Type BI [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Aspects for Usage Type DI [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guide for Usage Type MI [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Guide for Usage Type PI [SAP Library]</td>
</tr>
<tr>
<td>Usage Type AS</td>
<td>Interactive Forms based on Adobe Software Security Guide [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Internet Transaction Server Security [SAP Library]</td>
</tr>
<tr>
<td>Usage Type DI</td>
<td>Security of the SAP Java Development Infrastructure [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>The SAP NetWeaver Developer Studio: Security Aspects [SAP Library]</td>
</tr>
<tr>
<td></td>
<td>Security Aspects of Web Dynpro for Java [SAP Library]</td>
</tr>
</tbody>
</table>
Security Guides | Most-Relevant Sections
---|---
Connectivity and Interoperability | RFC/ICF Security Guide [SAP Library]
| Security Guide for Connectivity with the J2EE Engine [SAP Library]
| Web Services Security [SAP Library]

For a complete list of the available SAP Security Guides, see the Quick Link /securityguide on the SAP Service Marketplace at service.sap.com.

**Important SAP Notes**

The most important SAP Notes that apply to the security of the AS-Java are shown in the table below.

<table>
<thead>
<tr>
<th>SAP Note Number</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>724452</td>
<td>Central Note for SAP J2EE Engine 6.30/6.40</td>
</tr>
<tr>
<td>715371</td>
<td>Composite SAP note on security Basis 6.30/6.40</td>
</tr>
<tr>
<td>701385</td>
<td>SAP J2EE Engine 6.30, 6.40 Release Information</td>
</tr>
<tr>
<td>720590</td>
<td>User Management Engine (UME) on WAS 6.30 and higher.</td>
</tr>
<tr>
<td>871394</td>
<td>SAP NetWeaver 04s: Mandatory and optional J2EE Engine services</td>
</tr>
<tr>
<td>812332</td>
<td>How to set up logging on a remote J2EE client</td>
</tr>
<tr>
<td>710146</td>
<td>How to change J2EE Engine JVM Settings</td>
</tr>
<tr>
<td>723909</td>
<td>Java VM settings for J2EE 6.30/6.40</td>
</tr>
<tr>
<td>753002</td>
<td>Web Services SSL Proxy Authentication</td>
</tr>
<tr>
<td>764417</td>
<td>Information for troubleshooting of the SAP J2EE Engine 6.40</td>
</tr>
</tbody>
</table>

**Additional Information**

For more information about specific topics, see the Quick Links from SAP Service Marketplace.

**Quick Links to Additional Information**

<table>
<thead>
<tr>
<th>Content</th>
<th>Quick Link on the SAP Service Marketplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Related SAP Notes</td>
<td>/SAPNotesNW2004s</td>
</tr>
<tr>
<td>Security</td>
<td>/security</td>
</tr>
<tr>
<td>Security Guides</td>
<td>/securityguide</td>
</tr>
<tr>
<td>Network security</td>
<td>/network</td>
</tr>
<tr>
<td>Released platforms</td>
<td>/platforms</td>
</tr>
<tr>
<td>SAP Solution Manager</td>
<td>/solutionmanager</td>
</tr>
</tbody>
</table>
2 Technical System Landscape

For an overview of the relevant system components and communication paths in the technical system landscape of the AS-Java, see the figure below.

AS-Java Technical System Landscape

The complex communication patterns and links reflect the versatile functions that the AS-Java can perform in your system landscape. The AS-Java, and respectively the J2EE Engine, have an open and standards-based architecture that enables communication with a number of partners in your system landscape, for example Web clients, databases, AS-ABAP and other systems.

The versatility of the AS-Java and the number of communication partners, however, increase the system security risks, especially when used in open environments such as the Internet. Therefore, you can protect your productive systems by using firewalls to deploy the AS-Java system instances in a separate security zone in your system landscape. In addition, you can use application gateways, such as proxy servers, to filter and authenticate communication requests for the AS-Java and related application components.


You can also find more information from the resources listed below.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Guide/Tool</th>
<th>Quick Link to the SAP Service Marketplace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical description for the SAP NetWeaver AS Java</td>
<td>Master Guide</td>
<td>service.sap.com/instguides</td>
</tr>
<tr>
<td>High Availability</td>
<td></td>
<td>service.sap.com/ha</td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td>service.sap.com/security</td>
</tr>
</tbody>
</table>
3 User Administration and Authentication

For an overview of the user authentication and administration mechanisms for the AS-Java, see the following topics.

- **User Administration and Standard Users** [Page 3]
  Provides an overview of the required user types, the standard users that are delivered with the J2EE Engine, and the tools to use for user management.

- **User Data Synchronization** [Page 3]
  The AS-Java can use external data sources to store persistent user data. This topic describes how user data is synchronized with these other sources.

- **Authentication Mechanisms and Single Sign-On Integration** [Page 3]
  Describes the user authentication process on the J2EE Engine, the available authentication mechanisms and their integration in Single Sign-On environments.

See also:


3.1 User Administration and Standard Users

User Management Tools

The J2EE Engine allows you to manage user data and authentication parameters in a number of ways. For an overview of the user management tools and a comparison of the functions and tools you can use for remote user administration during server runtime, see **User Administration Tools** [Page 3].

User Types

Consistent with the open architecture of the J2EE Engine, user classification on the AS-Java follows group assignment and security policies. Using the user administration tools for the AS-Java, you can configure additional user types according to your authorization strategy and security needs.

For more information and standard user types, see **User Types** [Page 3].

Standard Users and Groups

The J2EE Engine has an open service provider architecture for storing user data, whereby you can use multiple user stores. The primary user stores on the J2EE Engine include the User Management Engine (UME) and the DBMS user store. In turn, the UME can use different user data sources, including AS-ABAP.

The standard users and groups are slightly different for each of the available user store options. You can use the standard users shipped with the J2EE Engine for each of the different user store options. Alternatively, you can use the user administration tools of the J2EE Engine to create additional users and user groups, according to your authentication and authorization priorities.

For a list of the standard users and user groups on the J2EE Engine, see **Standard Users** [Page 3] and **Standard User Groups** [Page 3].
### 3.1.1 User Administration Tools

The user administration tools for the AS-Java allow both offline and runtime user administration.

#### User Administration Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Detailed Description</th>
<th>Further Information</th>
</tr>
</thead>
</table>
| **Visual Administrator**            | A standalone graphical user interface (GUI) that enables local and remote management of users and groups at runtime and on the whole cluster, all cluster elements, and all modules running on them. | Administration Manual:  
  • Visual Administrator [SAP Library]  
  • J2EE Engine User Management Using the Visual Administrator [SAP Library] |
| **UME User Administration console** | A Web based tool that provides functions for managing users, groups, roles, and user-related data in the User Management Engine (UME). You can use the functions supplied with this tool from a Web client, such as a Web browser. | Administration Manual:  
  • User Management Administration Console [SAP Library] |
| **Shell Console Administrator**    | A command line tool that enables remote administration from Telnet clients. The default J2EE Engine configuration enables only Administrator users to use telnet. | Administration Manual:  
  • Shell Console Administrator [SAP Library]  
  • Connecting and Working Using Telnet [SAP Library] |
| **Config Tool**                     | An XML based tool that enables offline configuration of J2EE Engine cluster elements. Changes made using this tool must be exported to the engine database and take effect after engine restart. This tool does not support remote administration of J2EE Engines. | Administration Manual:  
  • Config Tool [SAP Library] |

### Remote User Administration During Runtime

User management during server runtime enables efficient and scalable user management for your productive systems. Furthermore, remote user administration facilitates security management of individual J2EE Engines, for example, when running in a cluster. Therefore, the J2EE Engine provides the following administration tools that allow remote user management during server runtime:

- Shell Console Administrator
- Visual Administrator
- UME User Management Console
For an overview and comparison of the available functions for these runtime user administration tools, see the table below:

<table>
<thead>
<tr>
<th>Function</th>
<th>Shell Console Administrator</th>
<th>Visual Administrator</th>
<th>UME User Management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create, view, or delete users</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Search for users</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Import users from external systems</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Replicate users to SAP systems</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Lock or unlock users</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>List locked users</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Change user passwords</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Define password rules</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Require password change</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Create, delete and manage groups and group members</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Assign a public-key certificate to a user</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Assign roles to users</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Assign UME actions to roles</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Change the user store</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

**See also:**

Administration Manual:
- [UME User Administration [SAP Library]]
- [J2EE Engine User Management Using the Visual Administrator [SAP Library]]
- [Shell Console Administrator [SAP Library]]
3.1.2 User Types

See the table below for the standard user types delivered with the J2EE Engine.

<table>
<thead>
<tr>
<th>User Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator users</td>
<td>Users who have unrestricted administration privileges over the J2EE Engine.</td>
</tr>
<tr>
<td>Guest users</td>
<td>Authenticated users with limited authorizations. Such users have limited system access or have registered as Company users in UME and are pending approval.</td>
</tr>
<tr>
<td>Authenticated users</td>
<td>Users who have been approved by a J2EE Engine administrator for dialog type access. For example, such users can be administered by the designated user administrators for their company.</td>
</tr>
<tr>
<td>Anonymous users</td>
<td>Users that can access the J2EE Engine without any form of authentication.</td>
</tr>
</tbody>
</table>

Communication and Service Users

The AS-Java is an application middleware component in your technical landscape. Therefore you can use it for communicating with various communication destinations and data sources. During the J2EE Engine installation, you create standard communication users to use for the connection between the J2EE Engine and the user data source server. The communication users are different, depending on the user store that you use. For more information and standard naming conventions, see Standard Users [Page 3] and Standard User Groups [Page 3].

After the AS-Java installation is complete or in your productive operations, you can use the J2EE Engine administration tools to customize the security configuration of the AS-Java for connecting to the various communication destinations in your system landscape. For more information, see the Optional Configuration [SAP Library] and Application Management [SAP Library] topics in the Administration Manual.

See also:

Administration Manual:

- UME User Administration [SAP Library]
- J2EE Engine User Management Using the Visual Administrator [SAP Library]
3.1.3 Standard Users

By default, the AS-Java provides standard users for administrative and guest access, as well as communication users for connecting to the installed user data store. The standard users on the J2EE Engine vary according to the user store and installation options and are as shown in the table below.

<table>
<thead>
<tr>
<th>Description</th>
<th>UME with AS-ABAP</th>
<th>UME with LDAP</th>
<th>Database (DB) store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator user</td>
<td>Specified during the installation. Example: J2EE_ADM_&lt;SID&gt;</td>
<td>Administrator</td>
<td>Administrator</td>
</tr>
<tr>
<td></td>
<td>For an add-in installation the standard user is J2EE_ADMIN.</td>
<td>This user has unlimited administrative permissions over the AS-Java. Therefore, we recommend that you set sufficiently strong password and auditing policies for this user.</td>
<td></td>
</tr>
<tr>
<td>Guest user</td>
<td>Specified during the installation. Example: J2EE_GST_&lt;SID&gt;</td>
<td>Guest</td>
<td>Guest</td>
</tr>
<tr>
<td></td>
<td>For an add-in installation, this user is J2EE_GUEST.</td>
<td>This user is also used for anonymous access to the AS-Java. By default, this user is locked.</td>
<td></td>
</tr>
<tr>
<td>Communication user</td>
<td>Specified during installation. Example: SAPJSF_&lt;SID&gt;</td>
<td>Configuration of the communication users for LDAP data sources is performed as an additional post-installation step. For more information, see Configuring UME to Use an LDAP Server as Data Source [SAP Library] in the UME documentation.</td>
<td>DB user is specified during installation. Example: SAP&lt;SID&gt;DB</td>
</tr>
<tr>
<td></td>
<td>In case you have several AS-Java systems with AS-ABAP data sources, we recommend that you create system-specific communication users using the above naming convention.</td>
<td></td>
<td>The J2EE Engine also uses this user for DB connectivity when you configure the UME with a DB user store.</td>
</tr>
</tbody>
</table>
When using the UME with AS-ABAP in an add-in installation, the AS-Java users must exist in the AS-ABAP data source. In addition, you have to complete the initial password setup for the AS-ABAP users, prior to creating the respective users on the AS-Java. For more information, see UME Data Sources [SAP Library] in the UME documentation.

In addition to the above standard users, a default AS-Java installation also contains the following service users:

<table>
<thead>
<tr>
<th>User</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADSuser</td>
<td>Used for communication between the J2EE Engine and the Adobe Document Services (ADS). This user is created in the J2EE Engine or in the AS-ABAP depending on the user store installation settings. For more information, see the Adobe Config Guide in the ADS Documentation and Interactive Forms Based on Adobe Software Security Guide [SAP Library] in the SAP NetWeaver Security Guide.</td>
</tr>
<tr>
<td>caf_mp_scvuser</td>
<td>Used internally in the Composite Application Framework (CAF) core transport system whenever the execution of a certain function requires administrator permissions, and the caller principal does not have this permission. The CAF also uses this service user to communicate with other J2EE Engine services. For more information, see Composite Application Framework Core Security Guide [SAP Library] in the SAP NetWeaver Security Guide.</td>
</tr>
</tbody>
</table>

### Security Restrictions for Standard Users

You assign initial passwords for the AS-Java standard users during installation. In your productive operations or after the installation is complete, you can use the User Management Engine (UME) and the AS-Java administration tools to change the initial passwords, manage the default properties for these users, lock users and create users with equivalent permissions on the J2EE Engine. For more information, see Managing Users [SAP Library] and UME User Administration [SAP Library] in the Administration Manual.

The Administrator user is required by certain applications on the J2EE Engine to perform administrative and installation tasks, for example software deployment and undeployment. Therefore, we recommend that you carefully evaluate changes to the role assignments for this user, protect it by defining strong authentication and password policies, and regularly audit its usage for accessing J2EE Engine resources.

### Emergency User

In case of emergency, you can enable the Emergency User store on the J2EE Engine. By default this user store contains only one user SAP*. For security purposes, when the Emergency User store is enabled, users defined in other user stores will be unable to access the J2EE Engine.

The SAP* user is the emergency user that has full administrative authorizations and can be used to reconfigure UME if the configuration is faulty and administrators and users can no longer access applications. To use this user, you must explicitly activate it and specify its password. For more information, see Activating the Emergency User [SAP Library] in the Administration Manual.
3.1.4 Standard User Groups

The standard user groups on the J2EE Engine are as shown in the table:

<table>
<thead>
<tr>
<th>User Group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrators</td>
<td>Contains all the users that have administrative privileges on the server or for the application. Users in this group have permissions to manage all other users (including other users with administrative privileges) as well as other security settings. No other users can perform user maintenance and security administration tasks.</td>
</tr>
<tr>
<td>Guests</td>
<td>Initially contains only the standard guest user (Guest or J2EE_GST_&lt;SID&gt;, depending on the user store).</td>
</tr>
<tr>
<td>Authenticated Users</td>
<td>Contains all non anonymous users, that is, users that have to authenticate themselves on the J2EE Engine.</td>
</tr>
<tr>
<td>Anonymous Users</td>
<td>Contains all named anonymous users that are listed in the ume.login.guest_user.uniqueids property in the UME properties.</td>
</tr>
<tr>
<td>Everyone (or all)</td>
<td>Contains all of the users and groups on the server.</td>
</tr>
</tbody>
</table>

⚠️

You should not create groups with the names of the groups Everyone, Authenticated Users, and Anonymous Users. If you create a group with one of these names through the native user interface of your LDAP directory or database, you will not get an error message, however, your user management will no longer function correctly. If you try to create a group with one of these names through the user management administration console, you will get an error message.

A default AS-Java installation also creates the user group ADSCallers for the service users of the Adobe Document Services (ADS), which are deployed on the J2EE Engine. For more information, see Interactive Forms based on Adobe Software Security Guide [SAP Library] in the SAP NetWeaver Security Guide.
Standard User Groups by User Stores

The standard user groups used by the different user stores on the AS-Java are as shown in the table below.

<table>
<thead>
<tr>
<th>Description</th>
<th>UME with AS-ABAP</th>
<th>UME with LDAP</th>
<th>Database User Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrators</td>
<td>Specified during the installation. Example: J2EE_ADMIN_&lt;SID&gt;</td>
<td></td>
<td>Administrators</td>
</tr>
<tr>
<td></td>
<td>For an Add-In installation this user is SAP_J2EE_ADMIN.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guests</td>
<td>Specified during the installation. Example: J2EE_GST_&lt;SID&gt;</td>
<td></td>
<td>Guests</td>
</tr>
<tr>
<td></td>
<td>For an Add-In installation this user is SAP_J2EE_GUEST.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Authenticated Users</td>
<td>Authenticated Users</td>
<td>Authenticated Users</td>
<td>Authenticated Users</td>
</tr>
<tr>
<td>Anonymous Users</td>
<td>Anonymous Users</td>
<td>Anonymous Users</td>
<td>Anonymous Users</td>
</tr>
<tr>
<td>Everyone</td>
<td>Everyone</td>
<td>Everyone</td>
<td>Everyone</td>
</tr>
</tbody>
</table>

3.2 User Data Synchronization

The J2EE Engine has an open service provider architecture for storing user data. In the standard system, SAP delivers the following user store providers:

- User Management Engine (UME) store provider
  
  The User Management Engine is the default active user store on the J2EE Engine. The UME itself has a number of options for storing user data. For more information, see UME Data Sources [SAP Library] in the UME documentation.

- DBMS user store provider
  
  This provider uses the internal database of the J2EE Engine as the store for the user data.

- UDDI user store provider
  
  Users that can publicize using the UDDI are stored in the UDDI user store and managed using the UDDI server management accessible form the Visual Administrator. In addition, the UDDI user store contains the credentials needed for using the UDDI Publishing API. For more information, see Managing the UDDI Server [SAP Library] in the Administration Manual and UDDI [SAP Library] in the Development Manual.
Overview of User Stores

You can configure several user stores in parallel, whereby you specify which user store is active in a server configuration. At runtime, the active user store is transparent to the user, and the user is not aware of the user store provider that is actually used for user authentication and authorization.

User Data Replication in UME

UME is the default user administration provider on the J2EE Engine. It enables scalable user administration, whereby you can replicate user data to external systems using a replication manager. User data can be replicated either automatically or, in extreme cases, manually. Consistent with the open architecture of user management in the AS-Java, UME also allows you to import and export user data from and to LDAP, database or AS-ABAP data sources.
For an overview of the architecture and the process flow of user data replication in the UME, see the figure below.

For more information and configuration details, see Concept of Interchangeable User Stores [SAP Library] and UME Configuration [SAP Library] in the Administration Manual. For reference material on the UME, including information on UME properties and configuration files, see UME Reference [SAP Library] in the Administration Manual.

See also:

### 3.3 Authentication Mechanisms and Single Sign-On Integration

The J2EE Engine implements the *Java Authentication and Authorization Service (JAAS)* standard to support various authentication methods. This enables you to choose the authentication mechanisms in your applications based on your authentication needs or requirements.

Applications running on the J2EE Engine can either use declarative or programmatic authentication. Both types of authentication rely on the same underlying technology, login modules and login module stacks. Programmatic authentication extends declarative authentication by using authentication schemes, which allow you to prioritize login module stacks and specify user interfaces for collecting authentication information.
SAP ships login modules and authentication schemes to support various authentication mechanisms. The following sections describe these concepts:

- **Declarative and Programmatic Authentication [Page 3]**
  
  Explains the difference between declarative (container-based) authentication and programmatic (UME) authentication. The type of authentication that an application uses has consequences for the login module stack it uses and on where you configure authentication.

- **Login Modules and Login Module Stacks [Page 3]**
  
  Here we provide conceptual information about login modules and login module stacks. Login modules define the authentication logic. Login module stacks enable you to define the sequence of authentication logic performed for an application.

- **Authentication Schemes [Page 3]**
  
  Here we provide conceptual information on authentication schemes.

- **Integration in Single Sign-On Environments [Page 3]**
  
  Here you can find an overview of the integration of the AS-Java authentication mechanisms in Single Sign-On environments.

**See also:**

**Administration Manual:**
- J2EE Engine User Management [SAP Library]
- Web Services Security [SAP Library]

**Development Manual:**
- Remote Authentication [SAP Library]
- Authentication for Web Applications Users on the J2EE Engine [SAP Library]

### 3.3.1 Declarative and Programmatic Authentication

Applications running on the J2EE Engine have two options for authenticating users:

- **Declarative authentication (also known as container-based authentication):**
  
  The Web container (in this case, the J2EE Engine) handles authentication. A component running on the J2EE Engine declares its protected resources and its desired authentication mechanism in its deployment descriptor. When a protected resource of this component is accessed, the container in which the component runs performs authentication.

- **Programmatic authentication (also known as UME authentication):**
  
  Components running on the J2EE Engine authenticate directly against the User Management Engine (UME) using the UME API. The component explicitly triggers authentication and then the authentication process is controlled by the authentication framework.
Both declarative and programmatic authentication use login modules and login module stacks as their underlying technology. Applications that use declarative authentication define which login module stack they use in their deployment descriptor. Programmatic authentication additionally introduces the concept of authentication schemes. Applications that use programmatic authentication are associated with an authentication scheme. The authentication scheme in turn references a login module stack. See also Login Modules and Login Module Stacks [Page 3] and Authentication Schemes [Page 3].

Integration

Different types of applications use different means of configuring which login module stack is used. The following table provides an overview.

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Type of Authentication</th>
<th>Where is Login Module Stack defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2EE Web applications</td>
<td>Declarative authentication</td>
<td>Declared in the web.xml deployment descriptor of the J2EE Web application. See Configuring Authentication [SAP Library].</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J2EE Web applications</td>
<td>Programmatic authentication</td>
<td>This depends on how the application is programmed. Applications can define an authentication scheme in their calls to the API. By default, if they do not define an authentication scheme, these applications use the login module stack referenced by default in the authentication schemes file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web Dynpro applications</td>
<td>Programmatic authentication</td>
<td>Web Dynpro applications always use the login module stack referenced by default in the authentication schemes file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Portal iViews</td>
<td>Programmatic authentication</td>
<td>An iView property defines which authentication scheme the iView uses. The authentication scheme references a login module stack. See Assigning an Authentication Scheme to an iView [SAP Library].</td>
</tr>
</tbody>
</table>

Declarative and programmatic authentication are integrated so that if an application uses programmatic authentication to authenticate its users, the container where it runs on the J2EE Engine is also aware that the users are authenticated. Inversely, if an application uses declarative authentication to authenticate its users, UME is also aware that the users are authenticated. Calls to the APIs of both the container and UME return the authenticated user.

If you wish to change the authentication mechanism used by Web Dynpro applications, you have to change the authentication scheme referenced by default in the authentication schemes file. If you change the default authentication scheme, this will also affect any portal iViews and J2EE Web applications that use the default authentication scheme.
3.3.2 Login Modules and Login Module Stacks

Login Module
Authentication on the J2EE Engine is performed using predefined authentication classes, referred to as login modules. A login module contains an implementation Java class that defines the authentication logic. For more information, see Login Modules [Page 3].

Login Module Stack
On the J2EE Engine you can use or define groups of login modules that contain different authentication logic. Such groups are referred to as login module stacks.

Each login module stack enables you to choose different combinations of authentication for applications you create or for each of the components on the J2EE Engine with applied security restrictions. For more information, see Login Module Stacks [Page 3].

Policy Configurations
The various components on the J2EE Engine, for example, J2EE applications, services, or modules, are registered with the Security Provider service so that you can apply security restrictions to them. The set of security restrictions for a component, which comprises of authentication and authorization rules, is referred to as a policy configuration.

Authentication Templates
You can use a login module stack for a component as an authentication template for other components. You can simply configure one login module stack and apply it to another registered component. For more information, see Managing Login Modules [SAP Library] and Managing Policy Configurations [SAP Library] in the Administration Manual.
Authentication on the J2EE Engine is performed using login modules. You can create your own login modules that implement and contain different types of authentication logic in them.

The J2EE Engine also provides a number of predefined login modules that contain authentication logic that defines different approaches to authenticating a client to the J2EE Engine. The following login modules are available:

<table>
<thead>
<tr>
<th>Login Module Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BasicPasswordLoginModule</td>
<td>Performs a JSP logon using Basic or Form authentication. That is, you use this login module to perform user authentication with user name and password. For more information, see Using Basic Authentication (User ID and Password) [SAP Library] in the Administration Manual.</td>
</tr>
<tr>
<td>CallerImpersonationMappingLoginModule</td>
<td>Used when the credentials of the caller principal are directly passed to the Enterprise Information System (EIS) and used for authentication of the resource principal.</td>
</tr>
<tr>
<td>ClientCertificateLoginModule</td>
<td>Performs a certificate logon to J2EE Engine. For more information, see Using Client Certificates for User Authentication [SAP Library] in the Administration Manual.</td>
</tr>
<tr>
<td>ConfiguredIdentityMappingLoginModule</td>
<td>Used when all caller principals obtain a connection to the EIS using use the same pre-configured identity. You have to specify either a user store that contains the identity, or a user name and a password for the configured identity.</td>
</tr>
<tr>
<td>CreateTicketLoginModule</td>
<td>Login module to create SAP Logon Tickets after successful logon. For more information, see Using Logon Tickets for Single Sign-On [SAP Library] in the Administration Manual.</td>
</tr>
<tr>
<td>CredentialsMappingLoginModule</td>
<td>Used when the credentials of the caller principal are replaced by the credentials that are used for authentication to the EIS; in this case, you have to specify a user store where the EIS credentials are stored.</td>
</tr>
<tr>
<td>CSILoginModule</td>
<td>Login module for the IIOP service.</td>
</tr>
<tr>
<td>DigestLoginModule</td>
<td>Authenticates applications, which defines their Digest authentication method in the deployment descriptors. This is a more advanced form of the Basic authentication type. Here the password of the user is digested (encoded).</td>
</tr>
</tbody>
</table>
## Login Module Stacks

### Definition

In the J2EE Engine you can use or define groups of login modules that contain different authentication logic. Such groups are referred to as login module stacks. Each login module stack enables you to choose different combinations of authentication for the applications you create, or for each of the components on the J2EE Engine.

### Login Modules

<table>
<thead>
<tr>
<th>Login Module Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EvaluateAssertionTicketLoginModule</td>
<td>Used for SSO with SAP Authentication Assertion Ticket. The login module verifies the ticket that arrives on the J2EE Engine.</td>
</tr>
<tr>
<td>EvaluateTicketLoginModule</td>
<td>Login module to evaluate SAP Logon Tickets.</td>
</tr>
<tr>
<td>HeaderVariableLoginModule</td>
<td>Login module at SSO using header variables. For more information, see Using Header Variables or Integrated Windows Authentication [SAP Library] in the Administration Manual.</td>
</tr>
<tr>
<td>PrincipalMappingLoginModule</td>
<td>Used when particular caller principals are mapped to an EIS principal. Only authorized caller principals can obtain a connection using a specific identity. You can either specify the user store where this identity is stored, or enter the name and the password of the resource principal.</td>
</tr>
<tr>
<td>SAMLLoginModule</td>
<td>Performs authentication using the SAML Browser/Artifact profile.</td>
</tr>
<tr>
<td>SecuritySessionLoginModule</td>
<td>Login module used by download.ear. It uses the tickets, generated by the Security Provider service on the engine.</td>
</tr>
</tbody>
</table>

You can combine these login modules in login module stacks. Such login module stacks can be used by the various components and applications on the J2EE Engine.

See also:
- Login Modules [SAP Library] in the Administration Manual
- Login Module Stacks [SAP Library] in the Administration Manual
Authentication Templates

Predefined login module stacks, which are also referred to as authentication templates, on the J2EE Engine:

- **SAP-J2EE-Engine** – this is a default configured login module stack that can be used by everyone.
- **Basic** – allows Basic Authentication, supported by the Web container.
- **Client** – allows client certificate authentication, supported by the Web container.
- **Digest** – allows digest authentication, supported by the Web container.
- **Form** – allows form authentication, supported by the Web container.
- **Ticket** – used for creating and verifying logon tickets.
- **Evaluation assertion ticket** – used for verifying assertion tickets (tickets used between systems).

Use

Using the login modules, you can create a login module stack that combines the authentication logic from several modules. Therefore, to perform authentication, the complete set of login modules is processed according to the configuration in the login module stack.

You can set the order in which these login modules are called during the authentication process and the order in which a client can be authenticated to the J2EE Engine. In addition, following the JAAS specification, each module is processed according to login module flags.

Login Module Flag Description

<table>
<thead>
<tr>
<th>Flag</th>
<th>Required to Succeed</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIONAL</td>
<td>No</td>
<td>Authentication proceeds down the list if the module has succeeded or has failed.</td>
</tr>
<tr>
<td>REQUIRED</td>
<td>Yes</td>
<td>Authentication proceeds down the list of modules if the module has succeeded or has failed.</td>
</tr>
<tr>
<td>REQUISITE</td>
<td>Yes</td>
<td>If successful, the authentication proceeds down the list, otherwise control returns to the application – that is, the authentication does not proceed.</td>
</tr>
<tr>
<td>SUFFICIENT</td>
<td>No</td>
<td>If the authentication is successful, control returns to application; otherwise, the authentication proceeds.</td>
</tr>
</tbody>
</table>
Example

The following table shows how a login module stack is processed based on these flags.

<table>
<thead>
<tr>
<th>Module</th>
<th>Flag</th>
<th>Pass/Fail</th>
<th>Pass/Fail</th>
<th>Pass/Fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module 1</td>
<td>SUFFICIENT</td>
<td>Pass</td>
<td>Fail</td>
<td>Fail</td>
</tr>
<tr>
<td>Module 2</td>
<td>REQUISITE</td>
<td>*</td>
<td>Pass</td>
<td>Fail</td>
</tr>
<tr>
<td>Module 3</td>
<td>OPTIONAL</td>
<td>*</td>
<td>Pass</td>
<td>*</td>
</tr>
<tr>
<td>Overall authentication</td>
<td></td>
<td>Pass</td>
<td>Pass</td>
<td>Fail</td>
</tr>
</tbody>
</table>

3.3.3 Authentication Schemes

Authentication schemes are used in UME authentication only. Portal iViews and Web Dynpro applications always use authentication schemes. J2EE Web applications may use them depending on whether they were programmed to use UME or container-based authentication. You only need to read this section if your applications use UME authentication.

Authentication schemes are defined in the authschemes.xml file, which you can change in the Config Tool. For more information, see Changing the authschemes.xml File [SAP Library] in the Administration Manual.

Definition

An authentication scheme is a definition of what is required for an authentication process. This includes:

- The login module stack that is used to determine whether a user is granted access to an application
- The user interfaces that are used to gather the information required to authenticate a user
- Priority, allowing authentication schemas to be ordered

Use

You use authentication schemes to define what type of authentication is required for a certain application. By assigning an authentication scheme to an application, you specify the type of authentication required for that application.

When users log on to an application and satisfy the authentication requirements for the authentication scheme required by the application, this information is stored in their logon ticket. If users try to access an application that requires a ‘stronger’ authentication scheme, the users will have to re-authenticate themselves and will be issued a new logon ticket with the new authentication scheme in it.

Authentication schemes enable pluggable authentication. You can easily ‘plug in’ additional authentication schemes without having to change each individual application.
Integration

The J2EE Engine is shipped with a set of authentication schemes. These are defined in the authschemes.xml file. For a list of the shipped authentication schemes, see Authentication Schemes Shipped with SAP NetWeaver AS Java [Page 3].

All Web Dynpro applications are automatically assigned to the default authentication scheme, which in turn references the ticket login module stack.

In the portal, each shipped iView template is assigned a reference to an authentication scheme. Initially all authentication scheme references point to the same authentication scheme (uidpwdlogon). If you have special authentication requirements, you can define custom authentication schemes and then change the configuration of the portal so that the references point to your custom authentication schemes. This allows you to change the authentication schemes without having to modify the iViews or iView templates.

⚠️ If you change the authentication scheme referenced by default, you automatically change the authentication scheme used by all Web Dynpro applications as well.

The following diagram illustrates this concept:

Initial Configuration of Portal

- iView Template A
  - AuthScheme=Default
  - uidpwdlogon (authentication scheme shipped with the portal)

- iView Template B
  - AuthScheme=UserAdminScheme

(Optional) Custom Configuration of Portal

- iView Template A
  - AuthScheme=Default
  - Custom authentication scheme for users

- iView Template B
  - AuthScheme=UserAdminScheme
  - Custom auth. scheme for user administrators
Authentication Schemes Shipped with SAP NetWeaver AS Java

The following authentication schemes are shipped with SAP NetWeaver Application Server Java:

<table>
<thead>
<tr>
<th>Name of Authentication Scheme</th>
<th>Description</th>
<th>Login Module Stack</th>
<th>Referenced by</th>
</tr>
</thead>
<tbody>
<tr>
<td>uidpwdlogon</td>
<td>Requires form-based logon with user ID and password.</td>
<td>ticket</td>
<td>default, UserAdminScheme</td>
</tr>
<tr>
<td>certlogon</td>
<td>Requires authentication using client certificates.</td>
<td>client</td>
<td></td>
</tr>
<tr>
<td>basicauthentication</td>
<td>Uses the Basic Authentication feature of the HTTP protocol.</td>
<td>ticket</td>
<td></td>
</tr>
<tr>
<td>header</td>
<td>Allows authentication using external Web access management products.</td>
<td>header</td>
<td></td>
</tr>
<tr>
<td>anonymous</td>
<td>Provides a very basic form of anonymous logon. A logon ticket is not issued.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.3.4 Integration in Single Sign-On Environments

The J2EE Engine is shipped with a range of login modules that support the most common authentication mechanisms. The majority of the supported authentication mechanisms enable efficient integration into Single Sign-On (SSO) environments.

For an overview of the authentication mechanisms shipped with AS-Java, see the table below:

<table>
<thead>
<tr>
<th>Authentication Mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Authentication</td>
<td>Basic Authentication is a HTTP standard method for authentication, whereby the user provides a user ID and password for authentication. By default, the J2EE Engine uses Basic Authentication for applications that are set up to use Basic or Form authentication.</td>
</tr>
<tr>
<td>Client certificates</td>
<td>As an alternative to Basic Authentication using a user ID and passwords, users using a Web browser as a front-end client can also provide X.509 client certificates to use for authentication. In this case, user authentication is performed on the Web server, no passwords have to be transferred and communication of user credentials is secured using the Secure Sockets Layer (SSL) Protocol. Authentication takes places without direct user intervention, which allows integration in a SSO environment.</td>
</tr>
<tr>
<td>Authentication Mechanism</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Security session IDs for Single Sign-On between applications</td>
<td>By default, the standard JSESSIONID mechanism is used to exchange the information about the user’s identity between the Web applications when performing SSO. SSO using the JSESSIONID mechanism, however, is only possible between applications that are running on the same server process in the cluster. This is because the security sessions created to identify the user’s security identity are not persistent and cannot be migrated to another server process. For this authentication mechanism, information about the user’s authentication and HTTP sessions is sent with the HTTP request using a session cookie (or URL rewriting, if cookies are disabled).</td>
</tr>
<tr>
<td>Logon tickets for Single Sign-On</td>
<td>The J2EE Engine supports the use of logon tickets for SSO when using a Web browser as the front-end client. In this case, users can be issued a logon ticket after they have authenticated themselves, using any of the supported mechanisms, with the initial SAP system. The ticket can then be submitted to other systems (SAP or external systems) as an authentication token. The user does not need to enter a user ID or password for authentication but can access the system directly after the system has checked the logon ticket.</td>
</tr>
<tr>
<td>Single Sign-On with resource adapters</td>
<td>The J2EE Engine provides an additional authentication mechanism type supported by the connector container: SAP Assertion Ticket of type com.sap.security.core.server.jaas.SAPAuthenticationAssertionTicketCredential. The SAP Assertion Ticket is fully compatible with the SAP Logon Ticket. With this authentication mechanism, you can specify additional authentication types in the deployment descriptor of the resource adapter according the J2EE Connector Architecture.</td>
</tr>
<tr>
<td>Header variables or integrated windows authentication for user authentication</td>
<td>You can use header variable authentication to delegate user authentication to any external product, which authenticates the user and returns an authenticated user ID as part of the HTTP header. Users only have to authenticate once against the external product and can then access applications on the AS-Java, such as the portal, with SSO. SSO with header variables can also be used for user authentication against an external Web Access Management (WAM) product. Both Windows NT LAN Manager (NTLM) and Kerberos authentication is supported. When using an external product with the header variable login module for authentication, all requests must pass through the external product. In addition, the user ID that the external product returns in the HTTP header must exist in the user management data sources.</td>
</tr>
</tbody>
</table>
SAML Assertions for Single Sign-On

<table>
<thead>
<tr>
<th>Authentication Mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAML Assertions for Single Sign-On</td>
<td>The J2EE Engine supports the use of the Security Assertion Markup Language (SAML) for SSO. You can use SAML for SSO in a scenario where a user is authenticated on an external authentication system that acts as an SAML authority. Based on this authentication, the user receives an SAML assertion (upon request) that he or she can use to access the J2EE Engine. To protect the data exchange, SSL is required for the connection between the source and destination sites. SSL is required by the SAML specification, and by default its use is activated in the SAML configuration. However, for testing purposes, you can disable the enforcement of SSL for the SAML-based document exchanges. The SAML Single Sign-On scenario involves a source site, assertion artifact and a destination site. The site that authenticates the user establishes a source site server that initiates the SAML communication. This source site provides the destination site with an assertion artifact, which is an identifier for the user's assertions. The source site also provides a responder, which acts as the SAML authority that actually provides the user's assertions. The destination site provides the desired resource and an artifact receiver, which receives the initial assertion artifact and passes it on to be evaluated by the SAML login module. If the user’s ID as provided by the SAML authority is not identical to the user ID at the destination site, then the destination site must also provide a user mapping mechanism.</td>
</tr>
</tbody>
</table>

If appropriate security measures are not taken, authentication using header variables can allow attackers to impersonate a user by sending a request with a user ID in the appropriate header variable to the J2EE Engine. To prevent this, you should make sure that the J2EE Engine can only be accessed from the authenticating Web server. If it is not possible to block the HTTP and HTTPS ports of the J2EE Engine, you must configure SSL with mutual authentication between the authenticating Web server and the J2EE Engine.

See also:

Administration Manual:

- Using Basic Authentication [SAP Library]
- Using Client Certificates for User Authentication [SAP Library]
- Using Security Session IDs for Single Sign-On between Applications [SAP Library]
- Using Logon Tickets for Single Sign-On [SAP Library]
- Using Single Sign-On with Resource Adapters [SAP Library]
- Using Header Variables or Integrated Windows Authentication for User Authentication [SAP Library]
- Using SAML Assertions for Single Sign-On [SAP Library]
4 Authorizations

Applications and components deployed on the J2EE Engine can use the following approaches to authorization checking:

- Assign activities to individual users based on roles
- Control the use of objects using Access Control Lists (ACLs).

In addition, applications deployed on the J2EE Engine can implement authorization checking using one of the following methods:

- Declarative - An application running on the J2 EE Engine declares its protected resources and the roles authorized for access. When a protected resource of this component is accessed, the container where the component runs performs the authorization based on this declaration.

- Programmatic – An application running on the J2EE Engine checks whether a caller of an Enterprise Java Bean or a Web resource has a specific role. In this case, the authorization check is specified in the component's application code and is carried out by the application and not the container where it is running.

Role Based Authorization Approaches

On the J2EE Engine, a security role represents an abstract logical group of users that is defined for a specific application components by the J2EE Engine administrator. The AS-Java can use two distinct types of role based authorization approaches to control access to protected resources.

- **J2EE roles**

  J2EE roles are security roles that are constructed according to the J2EE specification. They support both declarative and programmatic authorization checks. When using the declarative approach, the information is stored in deployment descriptors for the application. Alternatively, with the programmatic approach, the information is stored in the application code. To manage these roles you use the Visual Administrator tool of the J2EE Engine.

  J2EE roles are suitable for purely static, activity-related access control. This concept is based on the assignment of authorizations by activity (for example, the activity *Financial Accountant*), but not by instances (for example, by *Cost Centers*). For the examples, this means that all users assigned to the *Financial Accountant* role can post to all cost centers.

- **UME roles**

  The User Management Engine (UME) authorization concept extends the possibilities provided by the J2EE roles. UME roles are used by default on the J2EE Engine and it only supports a programmatic approach to authorization checking. To manage these roles you use the UME Administration Console of the Web Admin tool of the AS-Java.
The advantage of using UME roles is that administration is more flexible because you can consolidate permissions into actions, which are then grouped into roles and assigned to users. Therefore, the utilization of this concept allows you to assign authorizations both by activity and by instances. For example, this means that users assigned to the Financial Accountant role can only post to specific determined by you cost centers. In addition, when using UME roles, you can perform more complex checks than those supported when using J2EE security roles, such as using wildcards in the authorization check.

J2EE roles on the J2EE Engines are used purely declaratively. UME roles and the corresponding UME APIs are used for programmatic access control. For more information, see Architecture of Security Roles [SAP Library] in the Administration Manual.

Standard Roles and Actions

For more information about the standard UME actions and the roles for the AS-Java, see:

- Standard Roles [Page 3]
- Permissions, Actions and UME Roles [Page 3]
- Standard UME Actions [Page 3]

Access Control Lists

ACLs are suitable for protecting a large number of objects (that is, instances). In this case, you define an access control matrix that contains a subject (a role), a predicate (type of access), and the resource to be protected. Only users that are mapped to at least one of these roles can access this resource.

On the J2EE Engine, you use the Protection Domains and Policy Configuration in the Security Provider service to define ACLs. For more information, see Managing Protection Domains [SAP Library] and Managing Resources [SAP Library] in the Administration Manual.

See also:

Administration Manual

- Users and Authorization Administration in J2EE Engine [SAP Library]
- Resource Management [SAP Library]
- Security Roles Management [SAP Library]
- UME Roles [SAP Library]

Development Manual

- Using Security Roles and Security Role References [SAP Library]
- Specifying the Security Methods to Use for an Application [SAP Library]
- SAP NetWeaver Security Guide
- Deployment Authorizations When Using Deploy Service [SAP Library]
4.1 Standard Roles

Security roles on the J2EE Engine can be defined either globally or locally.

- **Globally for the J2EE Engine**
  You can use these security roles to protect any resources provided by services running on the J2EE Engine. These roles can be created automatically by a service, or manually by a J2EE Engine administrator user. You can access these roles from the policy configuration SAP-J2EE-Engine in the Security Provider service.

- **Locally for individual applications deployed on the J2EE Engine**
  The permissions for these roles are defined in the deployment descriptors of the deployed application. These roles have access permissions only for the applications that define them. Depending on the authorization method it uses, an application on the J2EE Engine can define a new security role or reference an existing global role.

You use the Visual Administrator tool of the J2EE Engine to manage J2EE roles. For more information, see [Security Roles Management](SAP Library) in the Administration Manual.

### Globally Defined Security Roles

The table below gives an overview of the globally defined security roles on the J2EE Engine.

<table>
<thead>
<tr>
<th>Role</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Security role used as a sum of all roles defined on the J2EE Engine.</td>
</tr>
<tr>
<td>Administrators</td>
<td>This role has unrestricted administrative permissions over the applications and services on the J2EE Engine.</td>
</tr>
<tr>
<td>Guests</td>
<td>This role has read-only permissions on the J2EE Engine.</td>
</tr>
<tr>
<td>KeystoreAdministrator</td>
<td>Users mapped to this role have full unrestricted access to all operations over all keystore views and entries. By default only the Administrators user group is mapped to this role</td>
</tr>
<tr>
<td>KeystoreViewCreator</td>
<td>This role has permissions to only create new keystore views. By default only the Administrators user group is mapped to this role.</td>
</tr>
</tbody>
</table>
## Locally Defined Security Roles

The table provides an overview of the local security roles used by the applications delivered with the J2EE Engine.

<table>
<thead>
<tr>
<th>Role</th>
<th>Application</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>view-creator</td>
<td>keystore-view:*</td>
<td>Has full unrestricted access to all operations over the current keystore view and its content. This role is mapped only to the creator user of the keystore view. If the view is created by a system thread, no user is mapped.</td>
</tr>
<tr>
<td>entry-creator.&lt;entry_name&gt;</td>
<td>keystore-view:*</td>
<td>Has full unrestricted access to all operations over the specified entry/property. The creator user of the specified entry/property is mapped to this role. If the specified entry/property is created by a system thread, no user is mapped.</td>
</tr>
<tr>
<td>entry-creator.&lt;property_name&gt;</td>
<td>keystore-view:*</td>
<td></td>
</tr>
<tr>
<td>telnet_login</td>
<td>service.telnet</td>
<td>Has access permissions for Telnet administration of the J2EE Engine.</td>
</tr>
<tr>
<td>Jndi_all_operations</td>
<td>service.jndi</td>
<td>Has permissions to perform all operations available in the naming system. If a user or a group does not have this permission assigned, it can only perform lookup operations.</td>
</tr>
<tr>
<td>Jndi_get_initial_context</td>
<td>service.jndi</td>
<td>Has permissions to receive InitialContext from the JNDI.</td>
</tr>
<tr>
<td>$SAP_J2EE_Engine.Upload</td>
<td>All applications that require upload permissions</td>
<td>Has upload permissions on the J2EE Engine. By default, components that require such permissions either create this role or use a reference to the globally defined Administrator role.</td>
</tr>
<tr>
<td>GRMGTTestServlet</td>
<td>com.sap.engine.heartbeat</td>
<td>Has permissions to use ping messages to ensure that the server connections are alive.</td>
</tr>
</tbody>
</table>
### Role Application Permissions

<table>
<thead>
<tr>
<th>Role</th>
<th>Application</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerGRMGApplication</td>
<td>com.sap.engine.heartbeat</td>
<td>Has permissions to send a GRMG request to a GRMG application, receive the GRMG response, display this response in a monitor in the alert monitor, send the response to the GRMG infrastructure, which displays it in the Alert Monitor there</td>
</tr>
<tr>
<td>XMLDASSecurityRole</td>
<td>com.sap/tc<del>TechSrv</del>XML_DAS*DataArchivingService</td>
<td>Security role with permissions for the XML Data Archiving Service Connector. For more information, see Administration of the XML Data Archiving Service [SAP Library] in the Administration Manual</td>
</tr>
<tr>
<td>LcrClassWriterAll</td>
<td>com.sap.lcr*sld</td>
<td>Has permissions to create, modify, and delete all types of CIM instances.</td>
</tr>
<tr>
<td>LcRCClassWriter</td>
<td>com.sap.lcr*sld</td>
<td>Has permissions to create, modify, and delete CIM classes.</td>
</tr>
<tr>
<td>LcRCClassWriterCR</td>
<td>com.sap.lcr*sld</td>
<td>Has permissions to create, modify, and delete CIM instances of the subset Component Information.</td>
</tr>
<tr>
<td>LcRCClassWriterLD</td>
<td>com.sap.lcr*sld</td>
<td>Permissions to create, modify, and delete CIM instances of the subset Landscape Description.</td>
</tr>
<tr>
<td>LcRCClassWriterNR</td>
<td>com.sap.lcr*sld</td>
<td>Permissions to create, modify, and delete CIM instances of the subset Name Reservation.</td>
</tr>
<tr>
<td>SAMLSSODemo_source</td>
<td>sap.com/tc<del>sec</del>app*samlssodemo_source</td>
<td>This role provides access protection to the source site servlet of the SAML SSO demonstration application delivered with the SAP NetWeaver AS Java.</td>
</tr>
<tr>
<td>SAMLSSODemo_responder</td>
<td>sap.com/tc<del>sec</del>app*samlssodemo_source</td>
<td>Provides access protection to the responder servlet of the SAML SSO demonstration application.</td>
</tr>
<tr>
<td>Role</td>
<td>Application</td>
<td>Permissions</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SAMLSSODemo_destination</td>
<td>sap.com/tc-sec-app* samlssodemo_dest</td>
<td>Provides access protection to the destination site servlet of the SAML SSO demonstration application.</td>
</tr>
<tr>
<td>ADSCaller</td>
<td>com.adobe/AdobeDocumentServices* AdobeDocumentServicesAssembly.jar</td>
<td>Role that allows the ADSuser to access the Adobe document services. For more information, see Adobe config guide.</td>
</tr>
<tr>
<td>Admin</td>
<td>com.adobe/AdobeDocumentServices* AdobeDocumentServicesAssembly.jar</td>
<td>Role with administrative privileges over the Adobe document services. For more information, see the Adobe config guide.</td>
</tr>
<tr>
<td>display</td>
<td>com.sap.rprof.RemoteProfile* exchangeProfile</td>
<td>Basic role for all users that are granted access to the remote profile application, used in the SAP Exchange Infrastructure.</td>
</tr>
<tr>
<td>SAP_JAVA_NWADMIN_CENTRAL</td>
<td>sap.com/tc-je-jmx-ws connector-app* sap.com/tc-je-jmx-wsconnector-ejb.jar</td>
<td>Has permissions to enable/disable certain elements in the UI of the Web Admin tool, as well as permissions to access the data in the managed system. This role can be used in the central system and the managed system as well. The central roles enable the management of the entire landscape that is available from SLD.</td>
</tr>
<tr>
<td>SAP_JAVA_NWADMIN_CENTRAL_READONLY</td>
<td>sap.com/tc-je-jmx-ws connector-app* sap.com/tc-je-jmx-wsconnector-ejb.jar</td>
<td>Read only permissions for the Web Admin tool. Does not allow any changes in the managed system such as start/stop, changing configuration.</td>
</tr>
</tbody>
</table>

See also:

- For information about standard SLD security roles, see the Post-Installation Guide – SLD on the SAP Service Marketplace at service.sap.com/SLD → Media Library
- Creating SLM Roles [SAP Library] in the Software Lifecycle Manager
- SAP NetWeaver Administrator Security Roles [SAP Library] in Administration Manual → System Landscape Administration with SAP NetWeaver
4.2 Permissions, Actions, and UME Roles

Definition

Authorizations are enforced in User Management Engine (UME) using permissions, actions, and roles.

Internally in their Java code, applications define Java permissions and use them for access control.

An action is a collection of permissions. Every application defines its own set of actions and specifies the permissions assigned to the actions either in an XML file or (more seldom) dynamically in the code. The actions are listed in the user management administration console, where you can group them together into roles.

UME Roles group together actions from one or more applications. You assign roles to users in the user management administration console. By assigning roles to users, you define the users’ authorizations.

Structure

The following figure illustrates the relationship between permissions, actions, and roles.

The advantage of having both actions and permissions is:

- Application developers can define finely grained permissions, but can hide the complexity by defining only a few actions.
- As the actions are normally defined in an XML file, they can be changed according to your requirements when you install the service.
- Administrators can assign actions to roles in the administration console. Permissions are not visible in the administration console.
Example

The user management administration console is an application running on User Management Engine. The application defines permissions in the code for activities such as changing a user’s profile or modifying roles. In the XML file an action `Manage_Roles` is defined, that groups together all permissions that a user requires to administrate roles. This action includes permissions for viewing, modifying, and deleting roles.

For example, you could create a role called `Role Administrator` and assign the action `Manage_Roles` to it. Then you could assign any administrator that requires permissions to administrate roles to the `Role Administrator` role.

Interfaces

The corresponding UME interfaces are included in the packages:

- `com.sap.security.api`
- `com.sap.security.api.acl`
- `com.sap.security.api.logon`
- `com.sap.security.api.ticket`

4.3 Standard UME Actions

The following table lists the UME actions delivered with the User Management Engine. These actions are defined in the file `UMErole.xml`.

<table>
<thead>
<tr>
<th>UME Action ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>UME.Manage_All</code></td>
<td>Provides permissions required by an overall user administrator. These include:</td>
</tr>
<tr>
<td></td>
<td>- Administration of users belonging to any company and possibility of assigning users to companies</td>
</tr>
<tr>
<td></td>
<td>(In a multitenant portal, even if a tenant user is assigned this action, he or she will still only have access to users, groups, and roles in his or her tenant.)</td>
</tr>
<tr>
<td></td>
<td>- Group management</td>
</tr>
<tr>
<td></td>
<td>- Role assignment</td>
</tr>
<tr>
<td></td>
<td>- User mapping</td>
</tr>
<tr>
<td></td>
<td>- Import and export of user data</td>
</tr>
<tr>
<td></td>
<td>- Manual replication of user data</td>
</tr>
<tr>
<td></td>
<td>To set up delegated user administration, overall user administrators must belong to a role to which the <code>UME.Manage_All</code> action is assigned.</td>
</tr>
<tr>
<td></td>
<td>In portal installations, any role that includes the <code>UME.Manage_All</code> action automatically has <code>Role Assigner</code> permissions on all portal roles in the portal installation.</td>
</tr>
<tr>
<td>UME Action ID</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>UME.AclSuperUser</strong></td>
<td>(Relevant for SAP Enterprise Portal only.) Provides Owner permissions on all objects in the Portal Content Catalog. It is not possible to remove this permission in the permission editor. This action is designed for super administrators.</td>
</tr>
<tr>
<td></td>
<td>This action should be used very restrictively as it provides extensive permissions on portal content. It should only be assigned to the Super Administration role in the portal. It should not be assigned to any other roles.</td>
</tr>
<tr>
<td><strong>UME.Manage_Users</strong></td>
<td>Provides permissions to administrate users belonging to the same company as the administrator (search, create, modify, delete, lock, unlock, reset password, approve new user requests).</td>
</tr>
<tr>
<td></td>
<td>To set up delegated user administration, delegated user administrators must belong to a role to which the <strong>UME.Manage_Users</strong> action is assigned.</td>
</tr>
<tr>
<td><strong>UME.Manage_Groups</strong></td>
<td>Provides permissions to view, add, modify, and delete groups and to assign users and groups to groups. Administrators can only assign users belonging to the same company as them.</td>
</tr>
<tr>
<td><strong>UME.Manage_Roles</strong></td>
<td>(Not relevant for SAP Enterprise Portal.) Provides permissions to view, add, modify, and delete UME roles, and to assign users and groups to UME roles.</td>
</tr>
<tr>
<td></td>
<td>Be careful to whom you assign this action. Users with this action can assign themselves the Administrator role which gives them full administrator rights on the J2EE Engine. In particular, DO NOT assign this action to delegated user administrators.</td>
</tr>
<tr>
<td><strong>UME.Manage_All_Companies</strong></td>
<td>Acts as an extension of UME actions so that they apply to users of all companies. For example, the action <strong>UME.Manage_Users</strong> provides permissions to administrate users belonging to the same company as the administrator. By additionally adding the action <strong>UME.Manage_All_Companies</strong>, the administrator has permissions to administrate users of all companies.</td>
</tr>
<tr>
<td><strong>UME.Sync_Admin</strong></td>
<td>Provides permissions to perform user replication using the user management administration console.</td>
</tr>
<tr>
<td><strong>UME.Batch_Admin</strong></td>
<td>Provides permissions to import and export users and groups using the user management administration console. Administrators can only import and export users belonging to the same company as them.</td>
</tr>
<tr>
<td><strong>UME.Manage_My_Profile</strong></td>
<td>Provides users with permissions to display and change their own personal user profile.</td>
</tr>
<tr>
<td></td>
<td>If the UME property ume.admin.allow_selfmanagement is set to true, this permission is not checked.</td>
</tr>
</tbody>
</table>
5 Network Security

Your network infrastructure is an important element in protecting your SAP NetWeaver systems. The network topology for the J2EE Engine is based on the topology used by the SAP NetWeaver platform.

SAP systems are implemented as client-server frameworks built in three levels: database server level, application server level and the presentation level (front ends). The AS-Java works at the database and application levels in your implementation framework, where it defines and forwards the presentation logic to Web client applications, such as, for example Web browsers and Web Dynpro applications.

Therefore, when deploying the AS-Java in your network landscape, we recommend that you use a network topology with multiple network security zones as shown in the figure below.

Network Topology with Multiple Security Zones

See the following sections for more details on the transport layer and communication channel security aspects that specifically apply to the AS-Java:

- Transport Layer Security [Page 3]
- Communication Channel Security [Page 3]
- J2EE Engine Ports [Page 3]

For a complete list of the default ports used by SAP NetWeaver products and their default assignments, see the document TCP/IP Ports Used by SAP Server Software, which is available on the SAP Service Marketplace at http://service.sap.com/security.
5.1 Transport Layer Security

For an overview of the communication protocols and the corresponding security mechanisms for the AS-Java, see the figure below.

**AS-Java Communication Paths and Protocols.**

Depending on the underlying communication protocol of the AS-Java, you can use either the Internet standard Secure Socket Layer (SSL) or Secure Network Communications (SNC). To utilize the transport layer security functions on the J2EE Engine, you must use an external security provider and the J2EE Engine must possess a security environment. You specify a security provider and the secure store security options during the AS-Java installation. For more information, see *Transport Layer Security [SAP Library]* in the Administration Manual.
For information specific to each of the communication protocols used by the AS-Java, see the table below.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Security Mechanism</th>
<th>Comment</th>
</tr>
</thead>
</table>
| HTTP         | Secure Socket Layer (SSL)                 | HTTP is the standard protocol to use for communication between Web clients, such as Web browser and Web applications, and the J2EE Engine. You can use SSL for authentication, integrity, and privacy protection.  
For more information, see Configuring the Use of SSL on the J2EE Engine [SAP Library] in the Administration Manual. |
| P4           | Secure Socket Layer (SSL)                 | P4 is the transfer protocol for Java specific Remote Method Invocation (RMI) communication and when using the Visual Administrator. This protocol is also used for communication between the SDM server and the J2EE Engine. P4 supports HTTP tunneling and can also be used with proxies.  
For more information, see Using P4 Protocol Over a Secure Connection [SAP Library] in the Administration Manual. |
| IIOP         | Secure Socket Layer (SSL)                 | IIOP is an alternative transfer protocol to use for RMI communication requests. You can also use IIOP for communication with CORBA application servers. Transport Layer Security for the IIOP protocol is provided by SSL.  
For more information, see Configuring the J2EE Engine for IIOP Security [SAP Library] in the Administration Manual. |
| LDAP         | Secure Socket Layer (SSL)                 | You can use an LDAP directory server as the persistence layer for the UME user store. You can use SSL for the Transport Layer Security in this case.                                                                 |
| RFC          | Secure Network Communication s (SNC)      | SNC is an SAP proprietary layer used with the SAP communication protocols RFC and DIAG.                                                                                                               
For more information, see Secure Network Communications (SNC) [SAP Library] in the SAP NetWeaver Security Guide |
| JDBC         | driver-dependent                          | JDBC is a communication protocol for connecting to databases. Transport Layer Security for database connectivity is provided by the driver used to connect to the database.                                      |
| Telnet       | Virtual Private Network (VPN)             | Telnet is used for remote administration using Shell Admin tool. We recommend establishing a virtual private network to secure the connection.                                                           
For more information, see Connecting and Working Using Telnet [SAP Library] in the Administration Manual |
Protocol | Security Mechanism | Comment
--- | --- | ---
SDM client/server TCP/IP based | Virtual Private Network (VPN) | The Software Deployment Manager (SDM) uses an SAP proprietary protocol based on TCP/IP for SDM client connection to the SDM server communication. To secure this communication channel, you need to use a Virtual Private Network.

Session | N/A | Session is a type of communication protocol that is used only between the dispatcher and server elements of the AS-Java cluster. These elements exist in the same security context of an AS-Java instance and, therefore, no security is necessary. For more information, see Cluster Communication [SAP Library] in the Architecture Manual.

See also:
Administration Manual:
- Configuring the Use of SSL on the SAP J2EE Engine [SAP Library]
- Using SSL to the SAP J2EE Engine via the ICM [SAP Library]
- Using SSL With an Intermediary Server [SAP Library]
- Configuring SNC (SAP J2EE Engine -> ABAP Engine) [SAP Library]

5.2 Communication Channel Security

Communication Flow
The J2EE Engine is an application middleware component in your system landscape and communicates with a number of communication partners. Deployed applications and the components of the J2EE Engine can negotiate communication using several protocols, depending on the J2EE Engine container where they reside. The primary communication protocols include HTTP, Telnet, P4 and IIOP, as well as, LDAP, JDBC and RFC for SAP specific communications. P4 and IIOP are the protocols that are used for Java specific Remote Method Invocations communication. In addition, the J2EE Engine supports SOAP for Web Services communication.

Communication Security

In the following topics we describe in more detail the relevant security considerations for each of the communication channels for the AS-Java:

- **Using and Intermediary Server to Connect to the J2EE Engine [Page 3]**
  Gives an overview of the security aspects associated with using intermediary devices such as the SAP Web Dispatcher, Microsoft IIS with IIS proxy module and other devices, for example, the Apache reverse proxy.

- **Communication Security for the Web Container [Page 3]**
  Presents an overview of the security aspects of the communication between the J2EE Engine and Web clients, for example Web browsers and Web Dynpro applications.

- **Communication Security for the EJB Container [Page 3]**
  Discusses the security aspects of the communication between application servers acting as clients or servers and the J2EE Engine.

- **Communication Security for Web Services [Page 3]**
  Provides an overview of the security aspects of the communication channels relevant to Web Services.

- **Communication Security for Persistence Layer [Page 3]**
  Provides an overview of the security aspects of connecting the J2EE Engine to backend systems and user persistency stores.

- **Communication Security for the Software Deployment Manager [Page 3]**
  Provides an overview of the communication channel security when deploying applications on the J2EE Engine using the Software Deployment Manager.

See also:

Data integrity

- **Specifying Security Constraints [SAP Library]** for J2EE applications in the Development Manual
- **Communication Security**
- **Security Guide for Connectivity with the J2EE Engine [SAP Library]** in the SAP NetWeaver Security Guide

### 5.2.1 Using an Intermediary Server to Connect to the J2EE Engine

When establishing your cluster network infrastructure, you can establish a demilitarized zone (DMZ) separated by using firewalls to control access to your more critical systems in the backend. Use an intermediary server in the DMZ to perform the necessary routing and load balancing tasks. See the figure below:
Using an Intermediary Server

Supported intermediary servers include:

- **SAP Web Dispatcher**
  The SAP Web dispatcher is a load-balancing device provided free-of-charge to SAP customers that operates using the SAP application server load-balancing mechanisms. It supports the use of SSL for securing connections and does have URL filtering capabilities. It can be used with both ABAP and J2EE installations.

- **Microsoft Internet Information Server (IIS) with the IIS proxy module**
  If you are using the Microsoft IIS as the intermediary server, you need to install and configure the IIS proxy module provided by SAP. This proxy module is an ISAPI filter that establishes the connection from the IIS to the J2EE Engine. It also supports the use of SSL for the connection to the J2EE Engine.

- **Other devices (for example, an Apache reverse proxy)**
  You can also use different devices as the intermediary server, for example, an Apache Web server as a reverse proxy. Such devices provide additional features such as more refined load-balancing, or request and content filtering. The features provided depend on the product you are using.

**See also:**

- [SAP Web Dispatcher](#)
- [Configuring the IIS as the Intermediary Server](#)
- [Configuring the Apache Web Server as the Intermediary Server](#)
5.2.2 Communication Security for the Web Container

For this communication channel, communication is initiated by a Web client, such as a Web browser or a Web Dynpro application. The access request coming from the Web client is passed through the AS-Java dispatcher for load balancing and is then forwarded to the Web applications (WARs) running in the Web container of the J2EE Engine. The Web applications then access business objects using Enterprise Java Beans (EJBs) from the EJB Container. The EJBs in turn access the actual data in the persistence layer.

For an overview of the communication flow, see the figure below.

Communication Flow for Web Container

The table below presents an overview of the security-relevant information for each of the communication paths.

<table>
<thead>
<tr>
<th>Communication Path</th>
<th>Protocol Used</th>
<th>Type of Data Transferred</th>
<th>Available Security Protection</th>
</tr>
</thead>
</table>
| Front-end client using Web client to application server | HTTP          | • Authentication information  
• All application data | Secure Socket Layer (SSL) |
| Web application to Enterprise Java Bean | P4 IIOP  | • All application data  
• Data about propagation of security credentials | Secure Socket Layer (SSL) |
| EJB to persistence layer | JDBC LDAP RFC  | • All application data  
• Authentication data when accessing persistence layers or remote servers | Driver dependent encryption for JDBC  
SSL for LDAP  
SNC for RFC |

See also: [Java Cluster Architecture (SAP Library)](SAP Library) in the Architecture Manual
5.2.3 Communication Security for the EJB Container

For this communication channel, communication occurs between RMI-P4, RMI-IIOP, or CORBA application servers acting as clients calling server-side remote objects such as Enterprise Java Beans (EJBs) or remote objects implementing RMI-P4 or RMI-IIOP.

![Communication Channel Diagram]

Application Server to Application Server Communication Flow

By contrast to accessing the J2EE Engine using Web applications, in this case, security management is carried out by the corresponding client or server side EJB container. The table below presents an overview of the security relevant information for each of the communication paths.

<table>
<thead>
<tr>
<th>Communication Path</th>
<th>Protocols Used</th>
<th>Type of Data Transferred</th>
<th>Available Security Protection</th>
</tr>
</thead>
</table>
| Client side RMI-P4 object accessing server-side EJB or remote object | P4             | • Authentication information  
• All application data                                              | Secure Socket Layer (SSL)             |
| Client side RMI-IIOP object accessing server-side EJB or remote object | IIOP           | • Authentication information  
• All application data                                              | Secure Socket Layer (SSL)             |
| Client side CORBA object accessing server-side EJB or remote object | IIOP           | • Authentication information  
• All application data                                              | Secure Socket Layer (SSL)             |
<table>
<thead>
<tr>
<th>Communication Path</th>
<th>Protocols Used</th>
<th>Type of Data Transferred</th>
<th>Available Security Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>EJB to persistence layer</td>
<td>JDBC, LDAP, RFC</td>
<td>• All application data&lt;br&gt;• Authentication data when accessing persistence layers or remote servers</td>
<td>Driver-dependent encryption for JDBC&lt;br&gt;SSL for LDAP&lt;br&gt;SNC for RFC</td>
</tr>
</tbody>
</table>

### 5.2.4 Communication Security for Web Services

A Web Service (WS) is a self-contained, modularized function that can be published, discovered, and accessed across a network using open standards. It represents an executable entity. For the caller or sender of a WS, a service is a black box that may require input and delivers a result. WS cover the provision of business integration functions within and across enterprises on top of any communication technology stack, whether synchronous or asynchronous.

The AS-Java uses the WS Framework for Java as a pluggable infrastructure for declaring and using Web Services. A Web Service can be made from any component, for example EJBs, Java Classes, Portal Services. The WS Framework takes care to deserialize incoming XML SOAP data and invoke an implementation. In addition, based on a Web Services Definition Language (WSDL) description, a WS proxy can be generated that exposes a Java Interface to the clients, and generates XML SOAP messages.

For an overview of the design time and runtime communication flow, see the figure below.
Web Services Communication Flow

To use a WS, a WS Consumer initiates a transaction with a WS provider using the Simple Object Access Protocol (SOAP). The SOAP transaction request is then transported over the network using the HTTP protocol. The transmission of the document can either be secured by using HTTP over SSL, or by signing and/or encrypting the SOAP document using OASIS WS Security.

Web service messages may travel over any number of connections and potentially traverse many intermediaries. In order to support this decoupled interaction, connection-oriented security, such as SSL, alone is insufficient or inappropriate. Therefore, the AS-Java enables you to use document security mechanisms, such as OASIS WS Security XML signatures and XML encryption, on a per message basis. In addition, to prevent unpredictable behavior of Web services due to poorly formed messages, with the AS-Java you can use a WS proxy.

You can use the AS-Java to act both as a provider and as a consumer for Web services. At design time you can use the SAP NetWeaver Development Studio provided for publishing, discovering, and accessing Web services on the AS-Java. Security-related features such as communication type or authentication level can be assigned in the WS definition. The technical details of these features are then specified in the WS configuration. WS definitions and deployed Web services are published in a UDDI registry using a WSDL document. During AS-Java system runtime, WSDL documents provide the basis for the WS consumer and can be retrieved from the UDDI using a browser or the standard UDDI APIs.

The AS-Java can provide both UDDI client and server functions. You can search in all, and publish to all, registries that conform to the UDDI standard. In addition, you can use the UDDI server, shipped as part of the AS-Java, to create your own registries. For more information, see Configuring the UDDI Client and UDDI Server [SAP Library] in the Development Manual.

The WS Consumer side derives the WS proxy generation based on the Web service definition, retrieved from the UDDI. Technical details that are predefined in the WS configuration are configured separately in the client runtime for the WS Container of the AS-Java. For more information, see Web Services Container Service [SAP Library] in the Administration Manual.
For an overview of the communication paths and the relevant security protection, see the table below.

<table>
<thead>
<tr>
<th>Communication Path</th>
<th>Protocol Used</th>
<th>Type of Data Transferred</th>
<th>Security Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS Consumption</td>
<td>SOAP over HTTP</td>
<td>WS application data in XML format. Authentication information</td>
<td>Secure Socket Layer. Document security</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• XML signature</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• XML encryption</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Client authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Client exclude lists when using a HTTP proxy server</td>
</tr>
<tr>
<td>Publish/Find WDSL</td>
<td>HTTP</td>
<td>WSDL application data</td>
<td>Secure Socket Layer. UDDI server Basic or Certificate Authentication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UDDI authentication information.</td>
<td>Client exclude lists when using a HTTP proxy server</td>
</tr>
</tbody>
</table>

See also:

Development Manual:
- [Web Services Security [SAP Library]]
- [Providing and Consuming Web Services [SAP Library]]
- [UDDI [SAP Library]]

Administration Manual:
- [Web Services Container Service [SAP Library]]
- [Web Services Security [SAP Library]]
5.2.5 Communication Security for Persistency Stores

As middle-tier software, the J2EE Engine communicates with a number of external data sources. The external data sources include SAP ABAP systems, databases, and LDAP-compliant directory servers.

For an overview of the communication flow, see the figure below.

Communication Flow for Persistency Stores

The table below presents an overview of the security-relevant information for each of the communication paths.

<table>
<thead>
<tr>
<th>Communication Path</th>
<th>Protocols Used</th>
<th>Type of Data Transferred</th>
<th>Available Security Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2EE Engine to LDAP directory server</td>
<td>LDAP</td>
<td>• Authentication data</td>
<td>Secure Socket Layer (SSL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data for directory management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Directory entries</td>
<td></td>
</tr>
<tr>
<td>J2EE Engine to RDBMS user store</td>
<td>JDBC 1.x</td>
<td>• Authentication data</td>
<td>Driver-provided encryption</td>
</tr>
<tr>
<td></td>
<td>JDBC 2.x</td>
<td>• Application data</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• User management data</td>
<td></td>
</tr>
<tr>
<td>J2EE Engine to SAP ABAP systems</td>
<td>RFC</td>
<td>• Authentication data</td>
<td>Secure Network Communications (SNC)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data</td>
<td></td>
</tr>
</tbody>
</table>

See also:

5.2.6 Communication Security for Software Deployment

Deployment is the final stage of application development and it involves the transfer of application data to the target system. You can use the following deployment tools for the AS-Java:

- **Software Deployment Manager (SDM) tool**
  The Software Deployment Manager (SDM) is a standard tool that you use to install Software and Development Components. The SDM is a client/server application that runs as a separate instance on the J2EE Engine side. You can use the SDM tool for the deployment of components on its internal file system, the J2EE Engine, and the databases the J2EE Engine uses. Remote communication, however, takes place only for the database and the J2EE Engine deployment targets.

  The SDM server is integrated in the Java Startup and Control Framework and can run either integrated with the J2EE Engine mode or as a standalone Java process. The SDM tool has three interfaces: a graphical user interface (GUI), a command line interface and a Java API, where only the GUI and the Java API can be used for client connections. For more information about the SDM, see [Software Deployment Manager (SAP Library)] in the Development Manual.

  The SDM server recognizes only one user, the SDM administrator, for client connections and only one user at a time can be logged on to the SDM. Therefore, it is important that you guard the password for the SDM administrator against unauthorized usage. In addition, SDM security-relevant data of the target SAP NetWeaver AS Java and Database systems is stored in the J2EE Engine’s secure store. This data is then accessed with the system ID (SID), which you can define in the SDM.

- **SAP NetWeaver Developer Studio**
  During software deployment, the SAP NetWeaver Developer Studio connects directly to the SDM server using the SDM Java API. Therefore, the security aspects of software deployment using the SAP NetWeaver Developer Studio are the same as for the SDM Java API.

- **Deploy Tool**
  This is a multipurpose tool that enables you to generate, assemble and deploy J2EE applications and application components. For more information, see [Deploy Tool (SAP Library)] in the Development Manual.

- **Visual Administrator**
  You can use the Visual Administrator tool’s deploy service to deploy J2EE applications, cluster elements and application components on the J2EE Engine.
For more information about using these tools in your deployment scenario, see Deployment: Putting It All Together [SAP Library] in the Development Manual. For an overview of the communication flow, see the figure below.

Communication Flow for Software Deployment

For an overview of the remote communication paths, the protocols used, the data transferred and the communication security for the SDM, see the table below.

<table>
<thead>
<tr>
<th>Communication path</th>
<th>Protocols Used</th>
<th>Type of Data Transferred</th>
<th>Available Security Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visual Administrator to J2EE Engine</td>
<td>P4</td>
<td>• Authentication data</td>
<td>• Secure Socket Layer (SSL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data</td>
<td>• HTTP/S tunnelling</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• J2EE application data for deployment</td>
<td></td>
</tr>
<tr>
<td>Deploy tool to J2EE Engine</td>
<td>P4</td>
<td>• Authentication data</td>
<td>• Secure Socket Layer (SSL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data</td>
<td>• HTTP/S tunnelling</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• J2EE application data for deployment</td>
<td></td>
</tr>
<tr>
<td>SDM remote GUI to SDM server</td>
<td>SAP proprietary protocol based on TCP/IP</td>
<td>• Authentication data for connection to SDM server</td>
<td>Virtual Private Network (VPN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data</td>
<td></td>
</tr>
<tr>
<td>SDM Java API-based client to SDM server</td>
<td>SAP proprietary protocol based on TCP/IP</td>
<td>• Authentication data for connection to SDM server</td>
<td>Virtual Private Network (VPN)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data</td>
<td></td>
</tr>
<tr>
<td>Communication path</td>
<td>Protocols Used</td>
<td>Type of Data Transferred</td>
<td>Available Security Protection</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------</td>
<td>----------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>SDM server to J2EE Engine</td>
<td>P4</td>
<td>• Authentication data for J2EE Engine connection</td>
<td>Secure Socket Layer (SSL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data for deployment</td>
<td></td>
</tr>
<tr>
<td>SDM server to database</td>
<td>JDBC</td>
<td>• Authentication data for database connection</td>
<td>Driver provided encryption</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Application data for deployment</td>
<td></td>
</tr>
</tbody>
</table>

See also:

Security of the SAP Java Development Infrastructure [SAP Library]
Working with the SDM [Page 3]
Deploy Service [SAP Library] in the Administration Manual

### 5.3 J2EE Engine Ports

These ports are used for communication with the J2EE Engine as part of the SAP NetWeaver AS. They are generated at installation time and when new cluster elements have to be created.

For the J2EE Engine, 20 ports are available for the dispatcher element and 80 are available for the server elements, since five is the maximum number of ports for a server process. Therefore, no more than 16 server cluster elements can be created on one instance.

The default J2EE Engine ports meet the following requirements:

- The port value is a number over 50000
- For each cluster element the ports begin with $50000+100 \times \text{instance\_number}$, where
  - instance\_number is a two digit number from 00 to 99 specifying the number of central instance and dialog instances
  - For dispatcher cluster elements
    - The ports are created from $50000+100 \times \text{instance\_number}+\text{port\_index}$, where
    - port\_index is a number from 0 to 19 inclusive.
Dispatcher port_index Values

<table>
<thead>
<tr>
<th>Index</th>
<th>Port Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>HTTP port</td>
</tr>
<tr>
<td>1</td>
<td>HTTP SSL port</td>
</tr>
<tr>
<td>2</td>
<td>IIOP Initial Context port</td>
</tr>
<tr>
<td>3</td>
<td>IIOP SSL port</td>
</tr>
<tr>
<td>4</td>
<td>P4 port</td>
</tr>
<tr>
<td>5</td>
<td>P4 HTTP Tunneling port</td>
</tr>
<tr>
<td>6</td>
<td>P4 SSL port</td>
</tr>
<tr>
<td>7</td>
<td>IIOP port</td>
</tr>
<tr>
<td>8</td>
<td>Telnet port</td>
</tr>
<tr>
<td>10</td>
<td>JMS port</td>
</tr>
</tbody>
</table>

According to the formula mentioned above and the port_index values, the dispatcher ports for the J2EE Engine are as follows (in the table below, NN corresponds to instance_number):

J2EE Engine Dispatcher Ports

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Port Number</th>
<th>Default Value</th>
<th>Range (min-max)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td>5NN00</td>
<td>50000</td>
<td>50000-59900</td>
</tr>
<tr>
<td>HTTP over SSL</td>
<td>5NN01</td>
<td>50001</td>
<td>50001-59901</td>
</tr>
<tr>
<td>IIOP</td>
<td>5NN07</td>
<td>50007</td>
<td>50007-59907</td>
</tr>
<tr>
<td>IIOP Initial Context</td>
<td>5NN02</td>
<td>50002</td>
<td>50002-59902</td>
</tr>
<tr>
<td>IIOP over SSL</td>
<td>5NN03</td>
<td>50003</td>
<td>50003-59903</td>
</tr>
<tr>
<td>P4</td>
<td>5NN04</td>
<td>50004</td>
<td>50004-59904</td>
</tr>
<tr>
<td>P4 over HTTP</td>
<td>5NN05</td>
<td>50005</td>
<td>50005-59905</td>
</tr>
<tr>
<td>P4 over SSL</td>
<td>5NN06</td>
<td>50006</td>
<td>50006-59906</td>
</tr>
<tr>
<td>Telnet</td>
<td>5NN08</td>
<td>50008</td>
<td>50008-59908</td>
</tr>
<tr>
<td>JMS</td>
<td>5NN10</td>
<td>50010</td>
<td>50010-59910</td>
</tr>
</tbody>
</table>

- For server cluster elements
  The ports are created from
  \[50000+100 \times \text{instance}_{-}\text{number}+20+n \times 5+\text{port}_{-}\text{index},\]
  where \(n\) is the number of server elements from 0 to 15 and \(\text{port}_{-}\text{index}\) is from 0 to 4.

  The ports for J2EE Engine server3 with \(\text{instance}_{-}\text{number}=15\) are from 51535
  \((50000+100 \times 15+20+3 \times 5+0)\) to 51539 \((50000+100 \times 15+20+3 \times 5+4)\).
According to the formula mentioned above and the port_index values, the server ports for the J2EE Engine are as follows:

### J2EE Engine Server Ports

<table>
<thead>
<tr>
<th>Internal Port</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Join Port</td>
<td>For s0: 5NN20, for s1=5NN25, for s2=5NN30,...for s15=5NN95, where s0, s1, s2,...s15 shows the number of server processes and NN is instance_number.</td>
</tr>
<tr>
<td>Server Debug Port</td>
<td>For s0: 5NN21, for s1=5NN26, for s2=5NN31,...for s15=5NN96, where s0, s1, s2,...s15 shows the number of server processes and NN is instance_number.</td>
</tr>
</tbody>
</table>

#### Example

The port for P4 on a dispatcher element with `instance_number=15` is:

P4 port=50000+100*15+4=51504

### 6 Data Storage Security

The J2EE Engine provides a secure storage area where applications or service components on the J2EE Engine can store sensitive data such as passwords or communication destinations, in encrypted form. Data saved in this area is encrypted using a secret key that is created explicitly for the application or service component.

#### Secure Storage in the File System

By default, the J2EE Engine stores secure data in the file \\usr\sap\<SID>\SYS\global\security\data\SecStore.properties in the file system. This file is created during installation and by default the J2EE Engine uses this file to store database connectivity information, such as the user SAP\<SID>DB, its password, the database pool, as well as, information for the user Administrator and its password.

The J2EE Engine uses the SAP Java Cryptography Toolkit to encrypt the contents of the secure store with the triple DES algorithm. During the AS-Java installation, you specify a key phrase that the J2EE Engine installation uses to generate the key that is used to encrypt the secure store data. After the installation is complete, you can use the J2EE Engine Config Tool to change the key phrase and re-encrypt the secure store. For more information, see Managing Secure Storage in the File System [SAP Library] in the Administration Manual.
Secure Storage for Application Specific Data

Applications or application components, deployed on the J2EE Engine, can save sensitive data in encrypted form in a secure storage area in the J2EE Engine’s configuration database. The data saved in this area is encrypted using a secret key that is created explicitly for the application or service. The J2EE Engine uses the triple DES algorithm to perform the encryption.

You can use two approaches for storing and maintaining the encrypted data for the individual applications or application components:

- **Centralized storage**
  
  With centralized storage, applications or application components use the Secure Storage service on the J2EE Engine to encrypt and decrypt the data. This data is also stored in the corresponding secure storage context on the J2EE Engine. You can control the parameters of this secure storage area from the properties of the Configuration Manager. For more information, see Configuration Manager [SAP Library] in the Administration Manual.

- **Decentralized storage**
  
  With decentralized storage, the applications and application component maintain their own storage area for the encrypted data. They only use the Secure Storage service on the J2EE Engine to retrieve the key, which is necessary to encrypt and decrypt the data.

Integration

Applications or services can use the J2EE Engine’s Secure Storage service to encrypt and store sensitive data such as passwords. To use the J2EE Engine secure store, applications or services are assigned a designated context area in the secure storage where the corresponding encrypted data is stored.

To receive a context area, J2EE Engine applications or services register with the secure storage service. The first time an application or service requests access to secure storage, the J2EE Engine registers the application or service, creates a context for the application or service, generates a secret key, and allows the application access to the context for future requests. For more information, see Secure Storage for Application-Specific Data [SAP Library] in the Administration Manual.

Software Deployment Manager Password Security

The Software Deployment Manager (SDM) uses the database connection information, the J2EE Engine administrator user and password from the secure storage in the file system, to connect to the J2EE Engine and perform tasks such as software deployment and undeployment.

⚠️

When you change the administrator's password in other user stores, you also have to update the password in secure storage by using the Config Tool. Otherwise, functions that use the administrator user, for example SDM, will not be able to authenticate to the J2EE Engine and deploy software components.
In addition, the SDM server stores deployment and configuration information in the SDM repository file. By default, this file is located in the SDM directory folder \usr\sap\<SID>\<instance_number>\SDM\program\config in the file system.

The SDM repository stores the password for the SDM administrator user as an irreversible hash. We recommend that you control access to the SDM repository file by setting appropriate access control permissions and use only the SDM GUI and the provided SDM command line interface to change the SDM configuration and the password for the SDM administrator. For more information, see Working with the SDM [Page 3] and the SDM documentation in the SDM folder in the file system.

See also:
Administration Manual:
- Changing the Administrator's Password and Updating it in Secure Storage [SAP Library]

7 Dispensable Functions with Impacts on Security

In productive mode, you should only have those services active on the J2EE Engine that are needed. For assistance on determining which services you need, see SAP Note 871394. This note provides a list of mandatory and optional services on the J2EE Engine in the AS-Java, based on the service references and independent of a specific use case.

The actual services needed depend on the applications that you are using on the J2EE Engine. Therefore, the list of services provided with the note is only a recommendation. To use this list on a productive system, you have to adjust the service classification according to the AS-Java system functions that you require for your specific use case.

The note contains several attachments, including an excel file about the AS-Java usage type of the SAP NetWeaver with several spreadsheets. For information about the contents of the worksheets, see the tables below.

Worksheets in Excel Table

<table>
<thead>
<tr>
<th>Name of Worksheet(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mandatory Services</strong></td>
<td>This worksheet lists the services that are mandatory for starting and running the J2EE Engine. The effects of disabling each service are also provided.</td>
</tr>
<tr>
<td><strong>Non-mandatory Services</strong></td>
<td>This worksheet lists the services on the J2EE Engine that are optional. The effects of disabling the services are also provided.</td>
</tr>
</tbody>
</table>
### Name of Worksheet(s) | Description
--- | ---
**Referencing Components** | This worksheet shows the hard and weak references that AS-Java service and interface components set to other deployed service and interface components.  
- For more information about J2EE Engine components and component references, see [J2EE Engine Components [SAP Library]](September 3) in the Architecture Manual.

**Referenced Components** | This worksheet shows the hard and weak references that AS-Java components receive from other service and interface components that are deployed on the AS-Java.

**Service – Interface Map** | This worksheet shows the AS-Java services, which explicitly declare the interfaces they provide on the J2EE Engine.

---

### 8 Other Security Relevant Information

The following topics provide an overview of additional security related information for the AS-Java.

- **Security on the JMS Service [Page 3]**
  
  In this topic we discuss the security aspects of the Java Message Service of the J2EE Engine. This service is used for exchanging messages between two or more Java clients. The security issues for this service that are discussed include authorization, authentication checking, policy configurations and communication protocols and ports.

- **Java Virtual Machine Security [Page 3]**
  
  The J2EE Engine runs in a Java Virtual Machine within your operating system. This topic gives an overview of the related security information.

- **Security Aspects of the Database Connection [Page 3]**
  
  The J2EE Engine uses the user persistence data stores provide for security and integrity of the data in cases of system upgrade or server failure. This topic gives an overview of the security mechanisms used for the integrity and confidentiality of the configuration and source code data stored in the user persistence stores of the J2EE Cluster.

- **Working with the SDM [Page 3]**
  
  The Software Deployment Manager (SDM) is a standard client/server tool that you use to install J2EE components on the J2EE Engine. In this topic, you can find security information related to using the SDM tool.
8 Other Security Relevant Information

- **Destination Service [Page 3]**
  Provides an overview of the security mechanisms in the Destinations service of the J2EE Engine. The Destination service is used by applications or services to specify the remote service's address and the user authentication information to use for connecting to other services.

### 8.1 Security on JMS Service

#### Security Aspects

The J2EE Engine JMS Provider implements the following security aspects:

- **Authentication**
  
  The `TopicConnectionFactory` is always obtained via JNDI. This is why the J2EE Engine forces authentication for JNDI clients. Only the clients that are authenticated against the JNDI login context can obtain a connection factory and create a JMS Connection. This user authentication is performed in one of the first steps you must follow to create a JMS Connection [SAP Library]. The JMS API provides the following two variants to make the authentication. You can:
  - Provide a user name and password:
    - The Security Provider Service of the session container forces authentication against the configured policy configuration for JMS basic password authentication.
  - Choose not to provide any credentials:
    - You must initialize the connection using SSL. The Security Provider Service of the session container then forces authentication against the policy configuration that is configured for JMS X.509 certificate authentication. Otherwise, the anonymous user is configured for J2EE Engine.

- **Authorization**
  
  J2EE Engine authorization of the JMS Connection enables you to apply restrictions over:
  - JMS Clients actions. You can prevent the client from:
    - Creating a `Producer` to a destination.
    - Creating a `Consumer` to a destination.
    - Creating a `Browser` to a destination.
    - Connecting to a `durable subscription on a destination`.
    - Creating a `durable subscription on a destination`.
    - Creating destination with `createQueue/createTopic`.
    - Creating a `temporary destination`.
  - JMS administrator actions
    - All administration tasks are checked against a predefined security role `JMSAdministrators` in the policy configuration of the Security Provider Service.
Authorization and authentication

Objects created for JMS using the Visual Administrator are stored in the JNDI. Such objects can contain user information such as passwords and if a user gains access to JNDI, then he or she can access the JMS configuration and other objects that have been created. We recommend that, in addition to protecting access (read, write, create) to the JMS service, you should restrict the access to the JNDI service using security roles.

`jms.authorization` defines the following resources:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Actions</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>queue</td>
<td>• consume</td>
<td>The resource for all queues. It has instances for each queue in the JMS Provider</td>
</tr>
<tr>
<td></td>
<td>• produce</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• browse</td>
<td></td>
</tr>
<tr>
<td>topic</td>
<td>• consume</td>
<td>The resource for all topics. It has instances for each topic in the provider. For hierarchical topics the instances are grouped</td>
</tr>
<tr>
<td></td>
<td>• produce</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• create-durable-subscription</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• connect-durable-subscription</td>
<td></td>
</tr>
<tr>
<td>subscription</td>
<td>• change-topic</td>
<td>The resource for all durable subscriptions</td>
</tr>
<tr>
<td></td>
<td>• change-selector</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• close</td>
<td></td>
</tr>
<tr>
<td>connection</td>
<td>• create-destination</td>
<td>The resource for client tasks that do not have to deal with a specific destination</td>
</tr>
<tr>
<td></td>
<td>• create-temporary-destination</td>
<td></td>
</tr>
<tr>
<td>administration</td>
<td>• create-topic</td>
<td>The resource for administrative tasks</td>
</tr>
<tr>
<td></td>
<td>• create-queue</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• create-subscription</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• remove-topic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• remove-queue</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• remove-subscription – if an</td>
<td></td>
</tr>
<tr>
<td></td>
<td>instance is used, it specifies</td>
<td></td>
</tr>
<tr>
<td></td>
<td>the target of the task</td>
<td></td>
</tr>
</tbody>
</table>
Policy configurations

The policy configurations for the JMS provider are created using the Security Provider Service interfaces. The following configurations are created:

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>jms.&lt;jms-instance-name&gt;.authentication.basic</td>
<td>Contains the authentication configuration for login using username-password. When it is created, it is set up from the policy configuration basic.</td>
</tr>
<tr>
<td>jms.&lt;jms-instance-name&gt;.authentication.cert</td>
<td>Contains the authentication configuration for login using X.509 certificate. When it is created, it is set up from the policy configuration client-cert.</td>
</tr>
<tr>
<td>jms.&lt;jms-instance-name&gt;.authorization</td>
<td>Contains restrictions for administrative tasks such as creating a destination as well as default client action restrictions. The JMS Provider has two security roles named JMSAdministrators, which reference the preconfigured security role administrators of the policy configuration SAP-J2EE-Engine, and JMSClients, which references the preconfigured security role guests of policy configuration SAP-J2EE-Engine. JMSAdministrators is granted to all administrative rights and JMSClients to all client actions. Additional security roles can be defined by an administrator.</td>
</tr>
</tbody>
</table>

Restrictions for message selectors

You can apply restrictions for the message selectors as follows:

For the instance of a destination, a number of message selection boundaries can be defined as grouped instances. When a consumer with a selector is created, the Security Service will go through all these boundaries to check if the consumer and selector combination passes all of them.
For a queue named `stocks` with two boundaries: a group of users `authorized` are allowed to view the messages on all stocks and another group called `not_authorized` is limited to stocks with IDs greater than 1000. Define two security roles then:

- `authorized` with the authorized users.
- `not_authorized` with the others users.

We define the following instances of resource `queue`:

- `queue stocks "stock-id <= 1000"` – authorized
- `queue stocks "stock-id > 1000"` – authorized, `not_authorized`

**Communication Protocols and Ports**

The JMS provider differentiates between internal and external communication.

- JMS internal communication is communication that takes place within the J2EE Engine cluster. Therefore, for internal communication both JMS and the application operate in the same runtime and therefore no extra security is necessary.

- External communication takes place using an SAP-proprietary binary format. The port used is obtained from the dispatcher. The default port is `5<sid>10`, however, you can change this port in the server port definitions. The protocol used for JMS can only be transferred using this port. When communicating over network boundaries, this port must be opened on the firewall.

> No encryption is available on the JMS communication. Even though the binary format is hard to decode this may imply a security risk if messages contain sensitive information.

**Data Storage**

Configuration data and user data (messages) are stored in the database and underlie the database protection mechanisms.
8.2 Java Virtual Machine Security

The J2EE Engine’s processes run in a Java Virtual Machine (JVM), which means that any security aspects that apply to the virtual machine also affect the security of the J2EE Engine.

Java’s security model defines the security concept and the mechanisms incorporated in the JVM. The Java security model is focused on protecting users from hostile programs downloaded from untrusted sources by providing a customizable sandbox in which Java programs run. The sandbox security model represents a shell that surrounds a running Java program and protects the host system from malicious code. Thus, because of the safety features defined by Java’s security model and incorporated in the JVM, running programs can access system resources only in safe and structured ways.

Java applications running in the JVM sandbox can also access native functions of the operating system where the sandbox runs. The security mechanisms of the JVM can establish whether a function can perform such access, however, they do not guard against malicious consequences from calling such native methods or software vulnerabilities in the application code of the JVM itself.

Therefore, we recommend that you follow the latest updates of JVM and install the latest patches provided by your virtual machine or operating system vendor.

8.3 Security Aspects for the Database Connection

Use

When connecting to the database, the J2EE Engine as well as the applications deployed on it authenticate themselves by means of a user name and a password. They are specified only once, when the DataSource that is used to provide the database connection is created. The DataSource is initialized with the supplied credentials and uses them for the authentication of all physical connections that it provides.

Features

You may use one of the following options for database connectivity:

- Using the default DataSource, you can connect to the system database in which the J2EE Engine stores its information
- You can register a new DataSource to connect to another database that your application uses

Using the Default DataSource

The default DataSource is created at installation and is used by all J2EE Engine services that need to connect to the system database. The applications that you later deploy on the server may also use this DataSource. For more information, see Using the Default DataSource [SAP Library].
The default DataSource uses the standard database schema user $SAP<\text{SID}>DB$, where $<\text{SID}>$ is the system identifier – for example, J2E. The password for this user is defined at installation.

The user name and password for the default DataSource, are stored encrypted in a secure storage. The parameters for this secure storage are the following properties of the Configuration Manager:

- `secstorefs.keyfile`
- `secstorefs.lib`
- `secstorefs.secfile`

For more information about these properties, see Configuration Manager [SAP Library] in the Reference Manual.

⚠️ You cannot establish a database connection and respectively run the J2EE Engine without using a secure storage. It is highly recommended that you do not change the default properties.

To change the password of the default user, you must:

- Change the user password in the database. For more information, see Database Access Protection [SAP Library].
- Maintain the relevant entry in the secure storage:
  - a. Start the Config Tool. (Execute the `configtool` script file in `<SAPj2eeEngine_install_dir>/configtool`.)
  - b. Select secure store. The configuration for the secure storage in the file system appears.
  - c. Select the `jdbc/pool/<\text{SID}>/Password` entry.
  - d. Enter the database user's new password in the Value field and choose Add.
  - e. Choose File → Apply to save the data. The new password is used to connect the J2EE Engine to the database the next time it is restarted.

See also:

Managing Secure Storage in the File System [SAP Library]

### Connecting with a User-Defined DataSource

If you need to connect to another database, you have to register a new DataSource using the JDBC Connector Service. For more information, see Creating a DataSource with JDBC 1.x Driver [SAP Library] and Creating a DataSource with JDBC 2.0 Driver [SAP Library].

To create the DataSource, you must supply a valid user name and password for the database schema. The J2EE Engine stores this data encrypted.
8.4 Working with the SDM

The Software Deployment Manager (SDM) is the standard tool that you use to install J2EE components on the SAP J2EE Engine. The SDM is a client/server application. The SDM Server runs on the SAP J2EE Engine side. This server is started automatically with the J2EE Engine. A graphical user interface is available as a client.

SDM Password

The SDM Server recognizes only one user, the SDM administrator. Anybody who has the password of this user can perform any activity in the SDM (such as deployment and undeployment). It is not easy to trace which user performed a particular activity.

To make access to the SDM server secure, change the default password as soon as possible after the installation. Use restrictive guidelines for the password. Also keep the group of people that knows the SDM password as small as possible. For more information, see Working with the Graphical User Interface (GUI) [SAP Library].

When you install SAP Web Application Server 6.40 (J2EE Stack) from Support Package 5, you have the option of entering a new password for the SDM directly, during the installation. You do not have this option if you upgrade with the patch procedure.

If your logon fails three times, the SDM Server (from SAP Web Application Server 6.40 SP5) stops automatically. You can then restart the SDM Server explicitly.

SDM Server

To prevent unauthorized persons from accessing the SDM Server, do not start it until you are installing J2EE components in the SAP J2EE Engine. Shut down the SDM Server again after the installation. For information on starting and stopping the SDM, see Starting and Stopping the Software Development Manager [SAP Library].

You can also use the command line interface of the SDM for deployments on the server host of the central instance of the SAP J2EE Engine. In this case, you do not need to start the SDM Server. For detailed information about the command line interface of the SDM, see the SDM installation directory (under <SDM-Install-Dir>/program/doc/SDMCommandLineDoc630_en_final.pdf).

As well as the deployment of software, you can also use the command line interface to perform administration tasks for the SDM Server, such as changing the password of the SDM Server. Any operating system users who can access the installation directory of the central instance can potentially perform any of these tasks. For this reason, carefully restrict access to the file system to trusted users.
Deployment with the SDM
To deploy J2EE components in the SAP J2EE Engine, use one of the following tools:

- SDM Remote GUI
- The command line interface of the SDM (local deployment on the central instance)
- Eclipse Deployment Plug-In

These tools all use the same communications protocol between the SDM server and SDM client. In all cases, the client-server connection is non-secure.

If you install sensitive components and data in central systems (such as a production server), you must make the connection secure by setting up a virtual private network (VPN) between the client and the server.

8.5 Destination Service

Use
Applications or services can establish connections to other services. When using such connections, you need to specify the remote service’s address and the user authentication information to use for the connection. Many applications use the Destination service for this purpose.

The Destination service supports the following types of destinations:

- HTTP(S)
- Web services
- RFC

You can manually create HTTP destinations and RFC destinations using the Destination service in the Visual Administrator.

Destinations for Web services are automatically created when specifying deployable Web service proxies (see Incorporating Web Services [SAP Library] in the Development Manual). However, you do have to enter the corresponding authentication information in the Web service destination.

Not all applications use the Destination service. Some applications have their own connection maintenance tools. For example, in the portal, you enter connection data in the system objects using the system landscape editor. Therefore, make sure you maintain the connection data in the correct location.
Integration
The Destination service uses the Secure Storage service on the J2EE Engine to store security-
relevant information such as passwords.

You can use the Secure Sockets Layer (SSL) protocol to secure HTTP connections. In this case,
the corresponding keys and public-key certificates are stored in keystore entries in the Key
Storage service.

You can use Secure Network Communications (SNC) to secure RFC connections to ABAP
systems. In this case, you need to use an external security product to provide the protection.

For more information, see Transport Layer Security on the SAP J2EE Engine [SAP Library].

Prerequisites
- If you use SNC to secure RFC connections, then the security product used is installed on
  the J2EE Engine. See Installing the SAP Cryptographic Library on the SAP J2EE Engine
  [SAP Library].
- The Destination, Secure Storage and Key Storage services must be running when an
  application or service requests access to its secure storage area.

Activities
For administrating the destinations, select the Destinations service in the Visual Administrator.
Under Runtime, the available destinations are displayed in the Destinations section. The
information that applies to a selected destination is displayed in the right pane.

For information about maintaining destinations, see:
- Maintaining HTTP and Web Service Destinations [SAP Library]
- Maintaining RFC Destinations [SAP Library]

Development
For more information about using the Destination service in your applications, see The
9 Tracing and Logging

Security tracing and logging are important elements for securing your application server systems. Therefore, the AS-Java includes monitoring and administration functions for the early detection and investigation of deviations from established security policies. The tracing and logging functions allow you to display security events for the entire J2EE Engine system landscape, as well as for individual J2EE Engines.

The logging and tracing functions on the J2EE Engine enable you to generate logs and traces for events affecting all components on the J2EE Engine. To facilitate information requirements for different levels of troubleshooting, logs are recorded by categories and traces by component location. In addition, you can use the administration tools of the J2EE Engine to generate logs and traces of system events with different severity levels.

The J2EE Engine uses the following categories to record system wide events:

- System - all system related security and administrative log messages.
- Applications - all system events related to the business logic.
- Performance - reserved for single activity tracing.

By default, the J2EE Engine saves logs and traces in files with extensions .log and .trc respectively. The location of these files in your file system is \usr\sap\<sid>\<instance_number>\j2ee\cluster\<node>\log

The following topics give additional information about the logging and tracing features for the AS-Java:

- [UME Logging and Monitoring](Page 3)
  Gives an overview of the logging and tracing files available for the User Management Engine of the J2EE Engine.

- [Security Aspects When Using HTTP and Web Container Tracing](Page 3)
  Presents the security aspects involved in enabling HTTP and Web Container tracing.

- [Masking Security Sensitive Data in the HTTP Access Log](Page 3)
  Includes information about the necessary configuration changes to mask security sensitive data written to the HTTP access logs.

The AS-Java monitoring and administration functions also enable you to view security events for the entire J2EE system landscape. For more information, see [Supportability and Performance Management](SAP Library) in the Administration Manual.

See also:

- [Log Configuration](SAP Library) in the Administration Manual
- [Logging and Tracing](SAP Library) in the Development Manual
9.1 UME Logging and Monitoring

The User Management Engine (UME) provides security logging which logs important security events such as successful and failed user logons, changes to users, user accounts, roles, and groups.

The following files contain the relevant logs and traces for important security events:

**Security Logging**

- **Location in Log Viewer:** ./log/system/security.log
- **Location in file system:** 
  /usr/sap/<SID>/<instance_number>/j2ee/cluster/server<X>/log/system/security.log

  This file contains a log of important security events, such as successful and failed user logons, and creation or modification of users, groups and roles.

**Security Tracing**

- **Location in Log Viewer:** ./log/defaultTrace.trc
- **Location in file system:** 
  /usr/sap/<SID>/<instance_number>/j2ee/cluster/server<X>/log/defaultTrace.trc

  This file contains all the trace information for the whole server and includes trace information for UME libraries and the UME Provider (com.sap.security.core.ume.service). The information in this file is very detailed and is mainly required by the SAP support team. Therefore, the security tracing includes exceptions, warnings, and debugging information.

The information in these files is accessible from the Log Viewer service in the Visual Administrator tool. In addition, the Security Provider service in the Visual Administrator tool allows you to monitor user logons and user sessions on the J2EE Engine.

**See also:**


9.2 Security Aspects When Using HTTP and Web Container Tracing

The HTTP Provider and Web Container services on the J2EE Engine provide options to configure the generation of traces that can be used for debugging problems with HTTP communication, or in Web applications that are deployed. Enabling tracing for any of these services can result in security-sensitive data being written to the corresponding trace files. Examples of security-sensitive data can be user credentials (username and password) being transmitted as request parameters, credentials that are passed as HTTP headers, the session ID being encoded in the URL, and so on.

HTTP and Web Container tracing is disabled by default. Therefore, you should carefully consider your security policy and the mechanisms your application uses to pass the security-sensitive data throughout the request-response cycle before enabling it. If you have debugged a problem and have used tracing for that, you have to apply security measures that prevent unauthorized access to information contained in it. By default, a certain level of security is provided as only users with administrative rights (at operating system level) can read files existing on the J2EE Engine’s file system.
Tracing HTTP Requests and Responses

You can enable tracing for HTTP communication using the `HttpTrace` property of the HTTP Provider Service as described in Enabling HTTP Traces [SAP Library]. The following table summarized the security risks associated with each of the values of this property:

<table>
<thead>
<tr>
<th>Property Value</th>
<th>Security Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>enable</code> or <code>enableHex</code></td>
<td>Security-sensitive information that is being passed as HTTP headers or with the body of the message is written to the trace file in plain text, or in hexadecimal format, respectively.</td>
</tr>
<tr>
<td><code>enableHeaders</code> or <code>enableHexHeaders</code></td>
<td>Only security-sensitive data that is passed as HTTP headers is traced.</td>
</tr>
</tbody>
</table>

Web Container Traces

You can enable tracing for the Web Container using the `TraceLevel` property of the Web Container Service as described in Enable and Configure Tracing for Web Applications [SAP Library]. In this case, any security-sensitive information that your application handles (passed as request/response attributes, stored in an HTTP session object, and so on) is written to the trace file.

9.3 Masking Security-sensitive Data in the HTTP Access Log

The HTTP Provider Service applies masking to the value of security-sensitive URL parameters, cookies, or headers that might be sent with the request. Those values appear as five dots in the relevant log file. The masking is applied for both Common Log File format, and the SAP log format that you might be using. For more information about log formats, see Logging in Common Log File Format [SAP Library].

HTTP headers values are not logged by default. The masking is applied only if you have configured the `LogHeaderValue` property of the HTTP Provider Service. For more information, see Logging Additional Information [SAP Library].

When using HTTP communication logging, you should consider your security policy, user access rights to log files and the mechanisms that deployed J2EE applications use to exchange security sensitive information over HTTP.

The J2EE Engine masks security-sensitive information in the HTTP communication logs as an additional step, based only on the parameters definitions and HTTP headers listed below. If you transmit security-sensitive information using custom parameters or custom defined headers, masking is not applied.
The following is a list of all elements masking applies to:

**URL Parameters**
- jsessionid
- mysapsso2
- j_password
- j_username
- j_sap_password
- j_sap_again
- oldPassword
- confirmNewPassword
- ticket

**HTTP Headers**
- Authorization
- Cookie
  - JSESSIONID
  - MYSAPSSO2

In addition, the J2EE Engine masks all HTTP data that was transferred via SSL.
10 Virus Protection and SAP GUI Integrity Checks

General Recommendations
The spreading of software viruses is not a new topic of discussion. Viruses have infected thousands of systems over the past several years and they should not be taken lightly. A successful virus infection can destroy data or block systems within moments. The spreading of viruses has been harmful enough on isolated machines where a single hard drive has been the storage medium. With the use of networks, the damage incurred by a virus infection increases enormously. An infected network can be rendered useless, costing a company money, time, and assets to re-establish itself. There are numerous virus-checking software packages on the market that you should use to check your software and you should update these regularly. Additionally, you need to educate your employees on the importance of being wary of unchecked software.

Using the Virus Scan Interface
The SAP Web Application Server has a virus scan interface that you can use to have data created or used by SAP software (or your own applications) checked for viruses. The interface itself consists of both an external interface for use with certified virus scan software vendors and an internal interface that you can use in your own developments. In addition, we provide a Business Add-In to integrate existing solutions.

For more information, see Virus Scan Interface [SAP Library].

SAP Software Virus Checks
We check the SAP software and data media for viruses before delivery. The SAP GUI for Windows presentation software also performs an integrity check on itself, which detects and reports any changes that may indicate a virus infection.